
**Road vehicles — Vehicle to grid
communication interface —**

**Part 4:
Network and application protocol
conformance test**

*Véhicules routiers — Interface de communication entre véhicule et
réseau électrique —*

Partie 4: Essai de conformité du protocole d'application et du réseau



STANDARDSISO.COM : Click to view the full PDF of ISO 15118-4:2018



COPYRIGHT PROTECTED DOCUMENT

© ISO 2018

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents	Page
Foreword	vii
Introduction	viii
1 Scope	1
2 Normative references	2
3 Terms and definitions	2
4 Symbols (and abbreviated terms)	7
5 Conventions	8
5.1 Requirement structure.....	8
5.2 Test system description	9
6 Test architecture reference model	9
6.1 General information	9
6.2 Platform adapter interface.....	10
6.3 SUT adapter interfaces	10
6.4 Codecs	11
7 Test suite conventions	11
7.1 General information	11
7.2 Test suite structure (TSS)	11
7.3 Test profiles.....	13
7.4 Test suite identifiers.....	62
7.5 Test suite coverage	67
7.6 Test case description.....	184
7.7 Test case specification	185
8 Test case descriptions for 15118-2 V2GTP	198
8.1 General information	198
8.2 SECC test cases	198
8.3 EVCC test cases.....	202
9 Test case descriptions for 15118-2 SDP messages	208
9.1 General information	208
9.2 SECC test cases	208
9.3 EVCC test cases.....	211
10 Test case descriptions for 15118-2 V2G application layer messages	236
10.1 General information	236
10.2 SECC test cases	236
10.3 EVCC test cases.....	405
Annex A (normative) Configuration specifications	671
A.1 Timer configuration	671
A.2 PICS configuration.....	671

A.3	PIXIT configuration	673
Annex B (normative) Control part specification.....		675
B.1	SECC control parts.....	675
B.1.1	AC specific control parts.....	675
B.1.2	DC specific control parts.....	675
B.2	EVCC control parts	688
B.2.1	AC specific control parts	702
B.2.2	DC specific control parts.....	702
Annex C (normative) Test-case specifications for 15118-2 V2GTP		745
C.1	SECC test cases.....	745
C.2	EVCC test cases	745
Annex D (normative) Test-case specifications for 15118-2 SDP messages		752
D.1	SECC test cases.....	752
D.2	EVCC test cases	752
Annex E (normative) Test-case specifications for 15118-2 V2G application layer messages.....		773
E.1	SECC test cases.....	773
E.1.1	V2G protocol handshake	773
E.1.1.1	SECC test cases for SupportedAppProtocol.....	773
E.1.2	V2G messages	773
E.1.2.1	SECC test cases for SessionSetup	775
E.1.2.2	SECC test cases for ServiceDiscovery.....	775
E.1.2.3	SECC test cases for ServiceDetail	782
E.1.2.4	SECC test cases for PaymentServiceSelection	784
E.1.2.5	SECC test cases for PaymentDetails	788
E.1.2.6	SECC test cases for Authorization.....	794
E.1.2.7	SECC test cases for ChargeParameterDiscovery	794
E.1.2.8	SECC test cases for PowerDelivery	799
E.1.2.9	SECC test cases for CertificateUpdate.....	824
E.1.2.10	SECC test cases for CertificateInstallation	839
E.1.2.11	SECC test cases for SessionStop.....	839
E.1.2.12	SECC test cases for ChargingStatus	842
E.1.2.13	SECC test cases for MeteringReceipt	846
E.1.2.14	SECC test cases for CableCheck	848
E.1.2.15	SECC test cases for PreCharge	852
E.1.2.16	SECC test cases for CurrentDemand	855
E.1.2.17	SECC test cases for WeldingDetection	859
E.2	EVCC test cases	860
Annex F (normative) Function specifications for supporting test execution		1009

F.1	Configuration functions.....	1009
F.2	Pre-condition functions.....	1009
F.2.1	SECC functions.....	1011
F.2.2	EVCC functions.....	1027
F.3	Post-condition functions.....	1027
F.3.1	SECC functions.....	1038
F.3.2	EVCC functions.....	1038
F.4	Common behavior functions	1039
F.4.1	SECC functions.....	1040
F.4.2	EVCC functions.....	1042
F.5	Library functions	1044
Annex G (normative)	Function specifications for 15118-2 V2GTP.....	1051
G.1	SECC functions.....	1051
G.2	EVCC functions	1053
Annex H (normative)	Function specifications for 15118-2 SDP messages.....	1056
H.1	SECC functions.....	1056
H.2	EVCC functions.....	1057
Annex I (normative)	Function specifications for 15118-2 V2G application layer messages	1061
I.1	SECC functions.....	1061
I.1.1	V2G protocol handshake	1061
I.1.1.1	SECC functions for SupportedAppProtocol	1061
I.1.2	V2G messages.....	1063
I.1.2.1	SECC functions for SessionSetup	1063
I.1.2.2	SECC functions for ServiceDiscovery	1064
I.1.2.3	SECC functions for ServiceDetail	1072
I.1.2.4	SECC functions for PaymentServiceSelection	1080
I.1.2.5	SECC functions for PaymentDetails	1087
I.1.2.6	SECC functions for Authorization.....	1091
I.1.2.7	SECC functions for ChargeParameterDiscovery	1104
I.1.2.8	SECC functions for PowerDelivery.....	1140
I.1.2.9	SECC functions for CertificateUpdate	1154
I.1.2.10	SECC functions for CertificateInstallation	1167
I.1.2.11	SECC functions for SessionStop	1176
I.1.2.12	SECC functions for ChargingStatus	1176
I.1.2.13	SECC functions for MeteringReceipt.....	1185
I.1.2.14	SECC functions for CableCheck	1197
I.1.2.15	SECC functions for PreCharge	1203
I.1.2.16	SECC functions for CurrentDemand.....	1207

I.1.2.17	SECC functions for WeldingDetection.....	1213
I.2	EVCC functions.....	1217
Annex J (normative)	Template specifications for V2G TCP/TLS Port Control	1373
Annex K (normative)	Template specifications for 15118-2 V2GTP.....	1375
K.1	Common templates.....	1375
Annex L (normative)	Template specifications for 15118-2 SDP messages.....	1376
L.1	Common templates.....	1376
Annex M (normative)	Template specifications for 15118-2 V2G application layer messages.....	1377
M.1	Common templates.....	1377
M.1.1	V2G protocol handshake	1378
M.1.1.1	CMN templates for SupportedAppProtocol	1378
M.1.2	V2G messages	1378
M.1.2.1	CMN templates for SessionSetup	1378
M.1.2.2	CMN templates for ServiceDiscovery	1378
M.1.2.3	CMN templates for Authorization	1381
M.1.2.4	CMN templates for PowerDelivery	1381
M.1.2.5	CMN templates for SessionStop	1383
M.1.2.6	CMN templates for ChargingStatus	1384
M.1.2.7	CMN templates for CableCheck	1384
M.1.2.8	CMN templates for PreCharge	1385
M.1.2.9	CMN templates for CurrentDemand	1385
M.1.2.10	CMN templates for WeldingDetection	1387
M.2	SECC templates	1387
M.3	EVCC templates	1397
Annex N (normative)	Template specifications for Security.....	1430
N.1	Common templates.....	1430
Annex O (normative)	Data type definitions.....	1434
O.1	Data types for PICS.....	1434
O.2	Data types for PIXIT	1434
O.3	Data types for V2G TCP/TLS Port Control	1436
O.4	Data types for V2GTP	1437
O.5	Data types for SDP messages.....	1438
O.6	Data types for V2G messages	1438
O.7	Data types for Security	1457

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html

This document was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 31, *Electrical and electronic equipment*, and Technical Committee IEC/TC 69 *Electric road vehicles and electric industrial trucks*. The draft was circulated for voting to the national bodies of both ISO and IEC.

A list of all parts in the ISO 15118 series can be found on the ISO website.

Introduction

The first three parts of ISO 15118 describe the use cases and the technical specification of the Vehicle-to-Grid Communication Interface which is intended for the optimized use of energy resources so that electric road vehicles can recharge in the most economic or most energy efficient way. It is furthermore required to develop efficient and convenient billing systems in order to cover the resulting micro-payments. The necessary communication channel may serve in the future to contribute to the stabilization of the electrical grid as well as to support additional information services required to operate electric vehicles efficiently and economically.

The complexity resulting from the network and application protocol requirements defined in the second part of the standard requires a considerable amount of testing in order to enable interoperability between independent implementations. This document therefore defines a conformance test suite for the network and application layer protocols in order to derive a common and agreed basis for conformance tests. The resulting test suite is a necessary prerequisite for downstream interoperability tests. Since interoperability furthermore involves the actual application logic of an implementation those tests are beyond the scope of this document. Hence this document focuses on the interface aspects and the corresponding requirements given in part two only.

STANDARDSISO.COM : Click to view the full PDF of ISO 15118-4:2018

Road vehicles — Vehicle to grid communication interface — Part 4: Network and application protocol conformance test

1 Scope

This document specifies conformance tests in the form of an Abstract Test Suite (ATS) for a System Under Test (SUT) implementing an EVCC or SECC according to ISO 15118-2. These conformance tests specify the testing of capabilities and behaviors of an SUT as well as checking what is observed against the conformance requirements specified in ISO 15118-2 and against what the supplier states the SUT implementation's capabilities are.

The capability tests within the ATS check that the observable capabilities of the SUT are in accordance with the static conformance requirements defined in ISO 15118-2. The behavior tests of the ATS examine an implementation as thoroughly as is practical over the full range of dynamic conformance requirements defined in ISO 15118-2 and within the capabilities of the SUT (see NOTE).

A test architecture is described in correspondence to the ATS. The conformance test cases in this document are described leveraging this test architecture and are specified in TTCN-3 Core Language for ISO/OSI Network Layer (Layer 3) and above. The conformance test cases for the Data Link Layer (Layer 2) and Physical Layer (Layer 1) are described in ISO 15118-5. Test cases with overlapping scopes are explicitly detailed.

This document does not include specific tests of other standards referenced within ISO 15118-2, e.g. IETF RFCs. Furthermore, the conformance tests specified in this document do not include the assessment of performance nor robustness or reliability of an implementation. They cannot provide judgments on the physical realization of abstract service primitives, how a system is implemented, how it provides any requested service, nor the environment of the protocol implementation. Furthermore, the test cases defined in this document only consider the communication protocol defined ISO 15118-2. Power flow between the EVSE and the EV is not considered.

NOTE 1 Practical limitations make it impossible to define an exhaustive test suite, and economic considerations can restrict testing even further. Hence, the purpose of this document is to increase the probability that different implementations are able to interwork. This is achieved by verifying them by means of a protocol test suite, thereby increasing the confidence that each implementation conforms to the protocol specification. However, the specified protocol test suite cannot guarantee conformance to the specification since it detects errors rather than their absence. Thus conformance to a test suite alone cannot guarantee interworking. What it does do is give confidence that an implementation has the required capabilities and that its behavior conforms consistently in representative instances of communication.

NOTE 2 This document has some interdependencies to the conformance tests defined in ISO 15118-5 which result from ISO/OSI cross layer dependencies in the underlying protocol specification (e.g. for sleep mode)

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61851-1:2017, *Electric vehicle conductive charging system — Part 1: General requirements*

ISO 15118-1:2013, *Road vehicles — Vehicle to grid communication interface — Part 1: General information and use-case definition*

ISO 15118-2:2014, *Road vehicles — Vehicle-to-Grid Communication Interface — Part 2: Network and application protocol requirements*

ISO 15118-3:2015, *Road vehicles — Vehicle-to-Grid Communication Interface — Part 3: Physical and data link layer requirements*

ETSI ES 201 873-5 V4.6.1, *TTCN-3: TTCN-3 Runtime Interface (June 2014)*

ETSI ES 201 873-6 V4.6.1, *TTCN-3: TTCN-3 Control Interface (June 2014)*

NOTE 1 Even though the technical specification ISO 15118-2:2014, which is the baseline for this conformance test document, explicitly references IEC 61851-1:2011, this document references IEC 61851-1:2017 because of applicability on the market.

3 Terms and definitions

For the purpose of this document, the terms and definitions given in ISO 15118-1, ISO 15118-2, ISO 15118-3 and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at <http://www.electropedia.org/>
- ISO Online browsing platform: available at <https://www.iso.org/obp>

3.1

abstract test case

complete and independent specification of the actions required to achieve a specific test purpose

Note 1 to entry: This specification is defined at the level of abstraction of a particular Abstract Test Method, starting in a stable testing state and ending in a stable testing state and may involve one or more consecutive or concurrent connections.

Note 2 to entry: The specification should be complete in the sense that it is sufficient to enable a test verdict to be assigned unambiguously to each potentially observable test outcome (i.e. sequence of test events).

Note 3 to entry: The specification should be independent in the sense that it should be possible to execute the derived executable test case in isolation from other such test cases (i.e. the specification should always include the possibility of starting and finishing in the “idle” state).

Note 4 to entry: Compare with ITU-T X.290.

3.2

abstract test suite

ATS

test suite composed of abstract test cases

Note 1 to entry: Compare with ITU-T X.290.

3.3**black box testing**

method of testing that examines the behavior of an SUT without considering the internal implementation and structure of the SUT, thus relying on the SUT's open interface for testing

3.4**conformance requirements**

conformance of a real system consisting of conformance to each requirement and conformance to the set

Note 1 to entry: Set of interrelated requirements which together define the behavior of the system and its communication. Conformance of a real system will, therefore, be expressed at two levels, conformance to each individual requirement and conformance to the set. Applicable ISO 15118-4 conformance tests include requirements and transfer syntax requirements as far as they can be validated by black box testing.

Note 2 to entry: See also *static conformance requirements* and *dynamic conformance requirements*.

3.5**conforming implementation**

IUT which satisfies both static and dynamic conformance requirements, consistent with the capabilities stated in the PICS(s)

Note 1 to entry: Compare with ITU-T X.290.

3.6**dynamic conformance requirements**

one of the requirements which specifies what observable behavior is permitted by the relevant specification(s) in instances of communication

Note 1 to entry: The requirements for this conformance specification are defined in ISO 15118-2.

Note 2 to entry: Compare with ITU-T X.290.

3.7**executable test case**

realization of an abstract test case

Note 1 to entry: Compare with ITU-T X.290.

3.8**expected behavior**

exact response of the SUT according to the underlying protocol specification to the stimulus defined in the test behavior

3.9**implementation conformance statement****ICS**

statement made by the supplier of an implementation or system claimed to conform to a given specification, stating which capabilities have been implemented

Note 1 to entry: The given specification for this conformance specification is ISO 15118-2.

Note 2 to entry: Compare with ITU-T X.290.

3.10**implementation extra information for testing****IXIT**

statement made by a supplier or implementer of an IUT which contains or references all of the information (in addition to that given in the ICS) related to the IUT and its testing environment, which will enable the test laboratory to run an appropriate test suite against the IUT

Note 1 to entry: Compare with ITU-T X.290.

3.11
implementation under test
IUT

implementation of one or more OSI protocols in an adjacent user/provider relationship, being that part of a real open system which is to be studied by testing

Note 1 to entry: Compare with ITU-T X.290.

3.12
main test component
MTC

single test component in a test component configuration responsible for creating and controlling *parallel test components* and computing and assigning the test verdict

Note 1 to entry: Compare with ITU-T X.290.

3.13
parallel test component
PTC

test component created by the main test component

Note 1 to entry: Compare with ITU-T X.292.

3.14
post-condition

test steps needed to define the path from the end of the *test behavior* up to the finishing stable state for the test case

Note 1 to entry: See also *Test behavior*.

3.15
pre-condition

test steps needed to define the path from the starting stable state of the test case up to the initial state from which the *test behavior* will start

Note 1 to entry: See also *Test behavior*.

3.16
protocol implementation conformance statements
PICS

ICS for an implementation or system claimed to conform to a given protocol specification

Note 1 to entry: The given protocol specification for this conformance specification is ISO 15118-2.

Note 2 to entry: Compare with ITU-T X.290.

3.17
protocol implementation extra information for testing
PIXIT

IXIT related to testing for conformance to a given protocol specification

Note 1 to entry: The given protocol specification for this conformance specification is ISO 15118-2.

Note 2 to entry: Compare with ITU-T X.290.

3.18

runtime environment

environment that describes the operating system and corresponding platform requirements of a system

EXAMPLE Test system.

3.19

semantically invalid test behavior

SemITB

test steps where the test system sends stimuli to the SUT that are semantically invalid according to the protocol requirements

Note 1 to entry: This type of test behavior is defined in this conformance standard and explicitly includes requirements which define the appropriate error handling of the SUT.

3.20

static conformance requirements

one of the requirements that specify the limitations on the combinations of implemented capabilities permitted in a real open system which is claimed to conform to the relevant specification(s)

Note 1 to entry: Compare with ITU-T X.290.

3.21

system under test

SUT

real open system in which the IUT resides

Note 1 to entry: Compare with ITU-T X.290.

3.22

syntactically invalid test behavior

SynITB

test steps where the test system sends stimuli to the SUT that are syntactically invalid according to the protocol requirements

Note 1 to entry: This type of test behavior is not defined in this conformance standard, see codec requirements.

3.23

test behavior

set of test steps (test body) which are essential in order to achieve the test purpose and assign verdicts to the possible outcomes

3.24

test execution

interpretation or execution of an abstract test suite

Note 1 to entry: Conceptually, the TE can be decomposed into three interacting entities: an Executable Test Suite (ETS), a Test Framework (TFW) and an optional internal Encoding/Decoding System (EDS) entity.

Note 2 to entry: See also ETSI ES 201 873-5 V4.6.1.

3.25

test framework

TFW

entity to perform all actions of test cases or functions

Note 1 to entry: The Test Framework interacts with the Test Management (TM), SUT Adaptor (SA) and Platform Adaptor (PA) entities via Test Control Interface (TCI) and Test Runtime Interface (TRI) and additionally manages the Executable Test Suite (ETS) and Encoding/Decoding System (EDS) entities. It initializes adaptors as well as ETS and EDS entities. This entity performs all the actions necessary to properly start the execution of a test case or function with parameters in the ETS entity. It queries the TM entity for module parameter values required by the ETS and sends logging information to it. It also collects and resolves associated verdicts returned by the ETS entity.

Note 2 to entry: See also ETSI ES 201 873-5 V4.6.1.

Note 3 to entry: In this document, the Test Framework TTCN-3 Runtime System (T3RTS) is used to explain a Test Framework functionality.

3.26

test purpose

prose description of a well-defined objective of testing, focusing on a single conformance requirement or a set of related conformance requirements as specified in the appropriate OSI specification

EXAMPLE Verifying the support of a specific value of a specific parameter

Note 1 to entry: Compare with ITU-T X.290.

3.27

test system

real system combining the test framework, abstract test suite, test execution and adapters as well as codecs

Note 1 to entry: Typically also containing a common runtime environment based on an operating system.

3.28

test control interface

TCI

four interfaces that define the interaction of the TTCN-3 Executable with the test management, the coding and decoding, the test component handling and the logging in a test system

Note 1 to entry: Compare with ETSI ES 201 873-6 V4.6.1.

3.29

test runtime interface

TRI

two interfaces that define the interaction of the TTCN-3 Executable between the SUT and the Platform Adaptor (PA) and the System Adaptor (SA) in a test system

Note 1 to entry: Compare with ETSI ES 201 873-5 V4.6.1.

3.30

test system interface

TSI

test component that provides a mapping of the ports available in the (abstract) TTCN-3 test system to those offered by a real test system

Note 1 to entry: Compare with ETSI ES 201 873-6 V4.6.1.

3.31

valid test behavior

VTB

test steps where the test system sends stimuli to the SUT that are valid (syntactically and semantically) according to the protocol requirements

Note 1 to entry: This type of test behavior is defined in this conformance document.

Note 2 to entry: The protocol requirements for this conformance specification are defined in ISO 15118-2.

3.32

verdict

test verdict

statement of “pass”, “fail” or “inconclusive”, as specified in an abstract test case, concerning conformance of an IUT with respect to that test case when it is executed

Note 1 to entry: Compare with ITU-T X.290.

4 Symbols and abbreviated terms

For the purpose of this document, the following abbreviations apply.

ALM	Application Layer Message
ATS	Abstract Test Suite
BEV	Battery Electric Vehicle
CA	Certificate Authority
CPL	Control Pilot Line
CRL	Certificate Revocation List
DH	Diffie Hellman
DER	Distinguished Encoding Rules
ECDSA	Elliptic Curve Digital Signature Algorithm
EDS	Encoding/Decoding System
EIM	External Identification Means
EMAID	E-Mobility Account Identifier
ETS	Executable Test Suite
EV	Electric Vehicle
EVCC	Electric Vehicle Communication Controller
EVSE	Electric Vehicle Supply Equipment
EXI	Efficient XML Interchange
HAL	Hardware Abstraction Layer
ICS	Implementation Conformance Statement
ID(s)	Identifier(s)
ITB	Invalid Test Behavior
IUT	Implementation under Test
IXIT	Implementation eXtra Information for Testing
OCSP	Online Certificate Status Protocol
OEM	Original Equipment Manufacturer

MTC	Main Test Component
NACK	Negative Acknowledgement
PDU	Protocol Data Unit
PICS	Protocol Implementation Conformance Statement
PIXIT	Protocol Implementation eXtra Information for Testing
PKI	Public Key Infrastructure
PLC	Power Line Communication
PnC	Plug and Charge
PTC	Parallel Test Component
PWM	Pulse-Width Modulation
SA	Secondary Actor
SDP	SECC Discovery Protocol
SECC	Supply Equipment Communication Controller
SLAC	Signal Level Attenuation Characterization
SUT	System Under Test
TC	Test Case
TCI	Test Control Interface
TCP	Transmission Control Protocol
TE	Test Execution
TFW	Test Framework
TP	Test Purpose
TRI	Test Runtime Interface
TSI	Test System Interface
TSS	Test Suite Structure
TTCN-3	Testing and Test Control Notation version 3
V2G	Vehicle-to-Grid
V2G CI	V2G Communication Interface
V2GTP	V2G Transfer Protocol
VTB	Valid Test Behavior
UDP	User Datagram Protocol
XML	Extensible Markup Language

5 Conventions

5.1 Requirement structure

This document uses unique number identifiers for each individual requirement. This requirement structure allows for easier requirement tracking and management. The following format is used throughout this document:

"[V2G"Y"- "XXX"]" requirement text

Where:

- "V2G" represents the ISO 15118 set of standards;
- "Y" represents the document part of the ISO 15118 document set, for this document Y = 4;
- "XXX" represents the individual requirement number; and
- "requirement text" includes the actual text of the requirement.

5.2 Test system description

TTCN-3 is used in this document to define/specify the test system and the test cases inside the test suite. However, using TTCN-3 is not mandatory for the implementation of conformance tests for ISO 15118-2.

[V2G4-001] The implementers of conformance tests shall verify that all test objectives implemented in their environment are identical to those described in this document.

6 Test architecture reference model

6.1 General information

Figure 1 provides an overview on the test architecture for this document. The following subclauses define the interface requirements marked black in Figure 1 for platform and SUT adapters (see 6.2 and 6.3) as well as the codecs (see 6.4). The Test Suite is defined in detail in the remainder of this document.

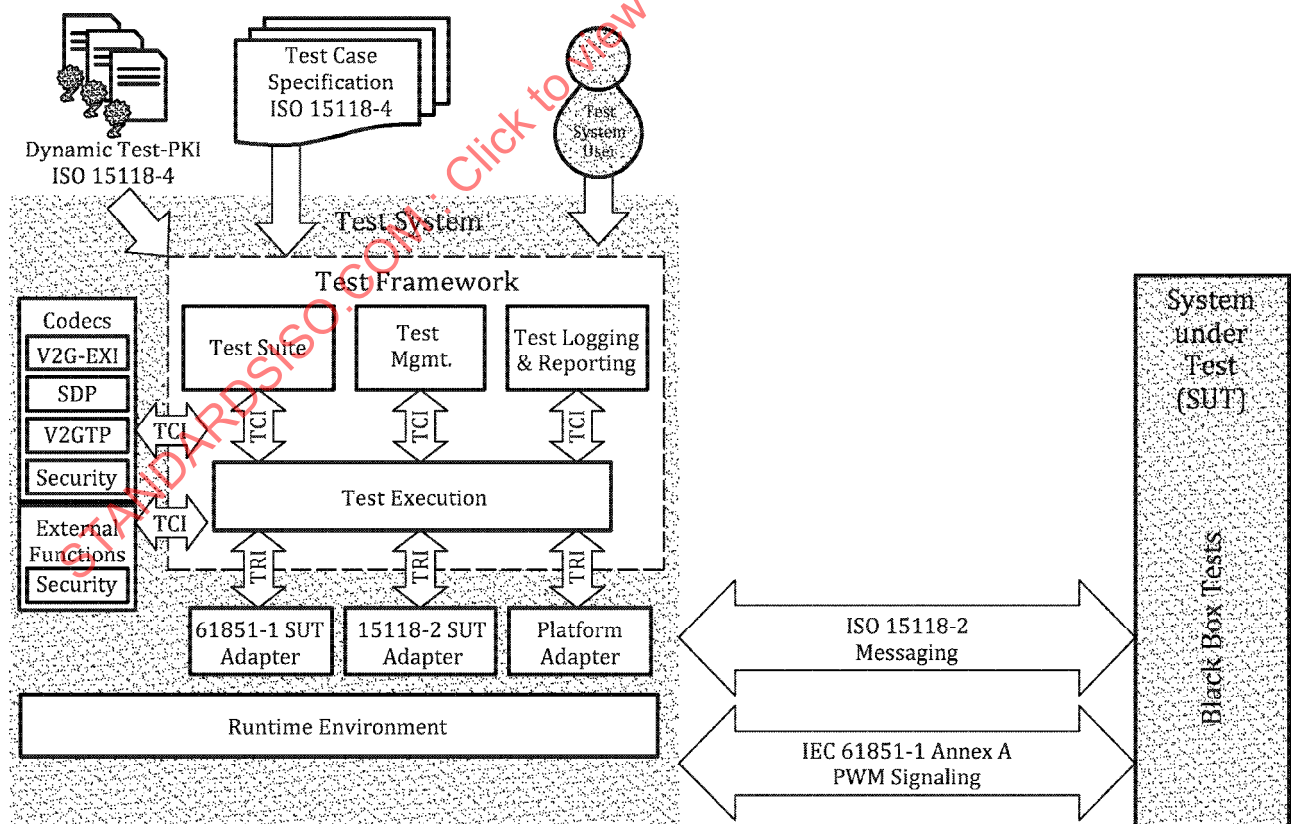


Figure 1 — Test architecture reference model for testing of ISO 15118-2

For the scope of this document only the "ISO 15118-2 – Messaging" interface to the SUT and its dependencies to the corresponding "IEC 61851-1:2017, Annex A – PWM Signaling" interface are relevant. This document therefore limits its scope to the respective SUT adapters and codecs.

6.2 Platform adapter interface

The platform adapter within the test system is responsible for timers and external functions. Besides means for timers, which are typically provided as part of the test framework, no external functions are defined for ISO 15118-2 conformance testing.

[V2G4-002] The platform adapter of the test system shall implement the TriPlatformPA and the TriPlatformTE interfaces as defined in ETSI ES 201 873-5 V4.6.1, 6.5.3.

6.3 SUT adapter interfaces

The SUT adapter within the test system adapts the TTCN-3 communication operations to the SUT based on an abstract test system interface and implements the real test system interface. It is responsible to propagate message requests and procedure based calls from the Test Execution (see Figure 1) to the SUT, and to notify the Test Execution of any received test events by appending them to its port queues.

[V2G4-003] The SUT adapters of the test system shall implement the TriCommunicationSA and the TriCommunicationTE interfaces as defined in ETSI ES 201 873-5 V4.6.1, 6.5.2.

NOTE 1 The actual implementation of these adapters is out of scope of this document.

[V2G4-004] The IEC 61851-1 SUT adapter shall adapt the HAL 61851-1 functions to a PWM Signal defined in 7.7.7.

NOTE 2 The IEC 61851-1 SUT adapter could need additional hardware for adapting the message-based HAL 61851-1 communication to PWM Signaling according to IEC 61851-1:2017, Annex A.

[V2G4-005] The ISO 15118-2 SUT adapter shall adapt and send/receive the encoded SDP requests/responses through UDP messages to/from the SUT as defined in ISO 15118-2:2014, 7.7.2 and 7.10.1.

[V2G4-006] The ISO 15118-2 SUT adapter shall adapt and send/receive the encoded V2G messages through a TCP or TLS connection to/from the SUT as defined in ISO 15118-2:2014, 7.7.1 and 7.7.3.

[V2G4-007] The ISO 15118-2 SUT adapter shall adapt the internal port control messages for TCP and TLS connection management as defined in Annex O, O.3

NOTE 3 The TCP/TLS connection management includes the following functionality:

- control for TCP/TLS port disconnection handling;
- control for receiving information/status on a given TCP/TLS connection;
- notification of TCP/TLS connection status events; and
- notification of TLS Handshake status events.

[V2G4-008] The ISO 15118-2 SUT adapter shall provide a port map operation (refer to 7.3.2) with additional parameters for TCP and TLS connection configuration as defined in Annex O, O.3.

NOTE 4 The TCP/TLS connection configuration includes the following functionality:

- configuration for IP address and port;
- configuration for usage of security (TCP/TLS);
- configuration for cipherSuite;

- configuration for TLS protocol version; and
- configuration for TLS certificate.

NOTE 5 The ISO 15118-3 adapter is defined in ISO 15118-5. It could need additional hardware to adapt between different Layer 2 communication technologies (e.g. Ethernet to HomePlug Green PHY).

6.4 Codecs

A codec is responsible for the external encoding and decoding of TTCN-3 values into bit strings suitable to be sent to the SUT. The Test Execution (TE) determines which codec shall be used and passes the TTCN-3 data to the appropriate encoder to obtain the encoded data. Received data is decoded in this entity by using the appropriate decoder, which translates the received data into TTCN-3 values cf. ETSI ES 201 873-6 V4.5.1, that can be matched against expected behaviors in the form of templates.

[V2G4-009] All codecs in this document shall implement the TCI-CD interface as defined in ETSI ES 201 873-6 V4.6.1, 7.3.2.

NOTE 1 For ISO 15118-2 conformance testing three codecs (V2G-EXI, SDP, V2GTP, see Figure 1) are used to encode or decode messages consumable by the tester into bit strings / signals consumable by the SUT.

NOTE 2 The exact implementation of these codecs is out of scope of this document.

[V2G4-010] The V2G-EXI codec shall encode TTCN-3 values as defined in Annex O, 0.6 into corresponding EXI streams as defined in ISO 15118-2:2014, 7.9.1.

[V2G4-011] The V2G-EXI Security codec shall encode TTCN-3 values for X.509 Certificates as defined in Annex O, 0.6 into a corresponding Base64 representation suitable for EXI streams according to ISO 15118-2:2014, 7.9.1.

[V2G4-012] The V2G-EXI codec shall decode EXI streams as defined in ISO 15118-2:2014, 7.9.1 into TTCN-3 values as defined in Annex O, 0.6.

[V2G4-013] The V2G-EXI Security codec shall decode Base64 representation from EXI streams according to ISO 15118-2:2014, 7.9.1 into TTCN-3 values for X.509 Certificates as defined in Annex O, 0.6.

[V2G4-014] The SDP codec shall encode TTCN-3 values as defined in Annex O, 0.5 into corresponding SDP requests/responses as defined in ISO 15118-2:2014, 7.10.1.3, 7.10.1.4 and 7.10.1.5.

[V2G4-015] The SDP codec shall decode SDP requests/responses as defined in ISO 15118-2:2014, 7.10.1.3, 7.10.1.4 and 7.10.1.5 into TTCN-3 values as defined in Annex O, 0.4.

[V2G4-016] The V2GTP codec shall encode TTCN-3 values as defined in Annex O, 0.4 into corresponding V2GTP envelopes as defined in ISO 15118-2:2014, 7.8.3.1.

[V2G4-017] The V2GTP codec shall decode V2GTP envelopes as defined in ISO 15118-2:2014, 7.8.3.1 into TTCN-3 values as defined in Annex O, 0.4.

7 Test suite conventions

7.1 General information

This clause defines all conventions that are relevant for conformance tests of SUTs implementing ISO 15118-2.

7.2 Test suite structure (TSS)

A test suite is a complete set of test cases, possibly combined into groups or modules (e.g. for use cases or domains like AC, DC charging), that are necessary to perform conformance testing for a given SUT.

In each test case, the SUT is stimulated with specific inputs and the reactions are observed and evaluated. Depending on the test purpose different pre-conditions and post-conditions shall be considered for the formulation of the test behavior. The pre-conditions, post-conditions as well as test behaviors are encapsulated into individual functions and stored within separate modules. Thus, a complete test case is composed by the actual test behavior enveloped by pre- and post-conditions (see 7.7.5 for details). The corresponding grouping of functions can therefore be assigned to the lowest abstract hierarchical level (see Figure 2). The test cases are defined on the second level.

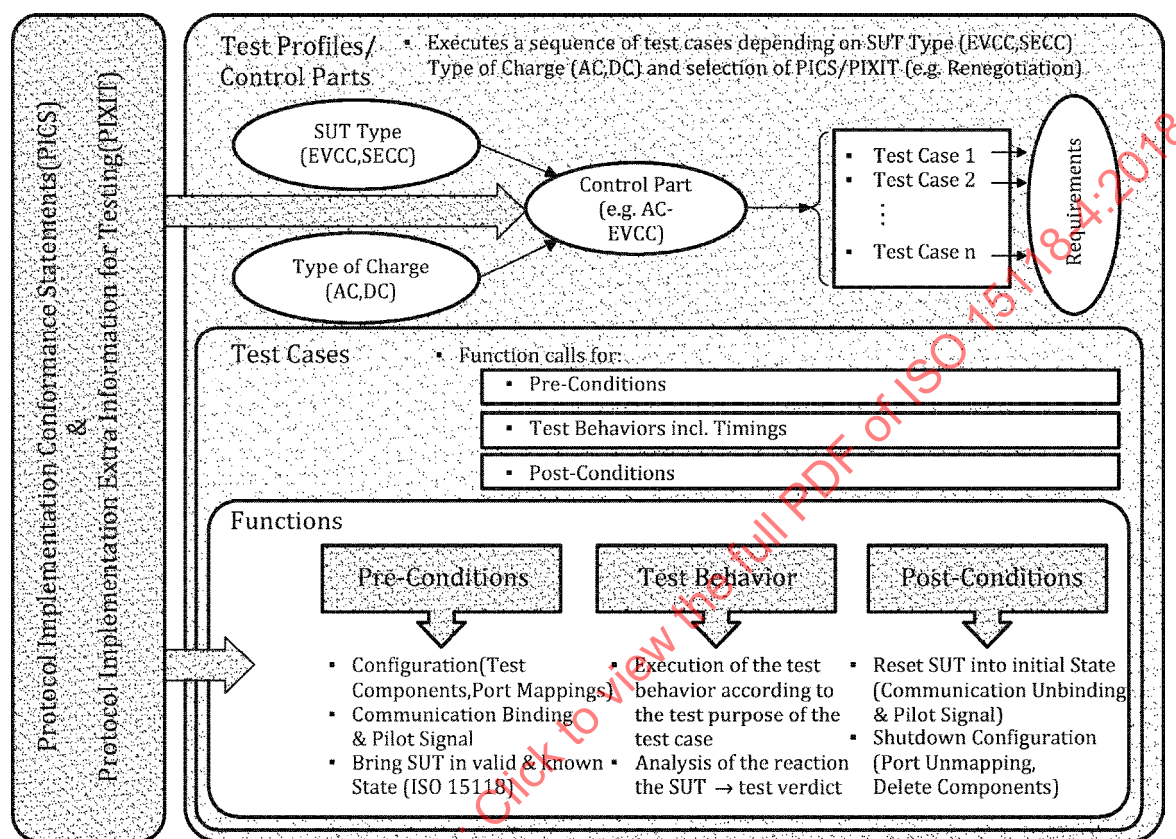


Figure 2 — General overview of the Test Suite Structure (TSS)

The test profile is a collection of self-contained test cases as well as PICS (see 7.3.3) and PIXIT (see 7.3.4) in order to represent a given use case. The selection is based on the use cases of the ISO 15118 standard and its corresponding requirements.

Hence the Test Suite Structure (TSS) is segmented into subgroups defined according to ISO 15118 use cases for conformance testing. Table 1 shows these subgroups, which are used for the organization of the test case specifications as well as for the test suite identifiers (see 7.4 for details).

Table 1 — Identifiers within the Test Suite Structure (TSS)

Identifiers	Values	Description
<sut>		System under Test
	EVCC	Electric Vehicle Communication Controller
	SECC	Supply Equipment Communication Controller
<dom>		Domain
	AC	AC specific behaviors
	DC	DC specific behaviors
	IN	Inductive specific behaviors

	CMN	Common behaviors
<ctx>	{fullname}	Context (e.g. name of message pattern signal name according to standard)

NOTE 1 The domain for inductive charging is not relevant for this document but was introduced for future purposes.

Table 2 describes the Annexes derived for this document.

Table 2 — Test suite structure – Annexes description

Annexes	Description
Configuration Annex A	The configuration annex contains constant definitions timers, PICS and PIXIT definitions. The configuration described in Annex A shall be applied.
Control Part Annex B	The control part annex describes the execution order of modeled test cases (test groups). The EVCC control part subclauses contain the test groups if the SUT is an EVCC. The SECC control part subclauses contain the test groups if the SUT is an SECC. The control part in Annex B shall be followed.
Test Cases Annex C - E	The test case annexes contain the TTCN-3 test case files. The EVCC subclauses contain the test cases if the SUT is an EVCC. The SECC subclauses contain the test cases if the SUT is an SECC. The control part in Annexes C to E shall be considered.
Functions Annex F – I	The function annexes contain methods for supporting the test execution as well as the actual pre-, post-conditions and test behaviors of the ISO 15118 conformance tests. The EVCC subclauses contain the test behavior if the SUT is an EVCC. The SECC subclauses contain the test behavior if the SUT is an SECC. The Pre-condition subclause contains functions that are used for defining the pre-conditions of a test behavior. The Post-condition subclause contains functions that are used to establish the post-conditions of a test case. The Library Functions subclause contains utility functions used across the test suite for various purposes. The functions in Annexes F to I shall be considered.
Templates Annex J - N	The Templates subclause contains the TTCN-3 template files for matching a SUT's reaction on a stimulus against its expected behavior. The templates in Annexes J to N shall be considered.
Data Structures Annex O	The Data Structures subclause contains data structures that are needed additionally to the ISO 15118 data structures e.g. used for X.509 Certificates. The data structures in Annex O shall be considered.

7.3 Test profiles

This subclause defines test profiles for conformance with ISO 15118-2. A test profile consists of a test configuration as well as a selection and assignment of PICS/PIXIT. Depending on the test configuration a set of test components and ports are defined. The test profile furthermore includes a test group defining the set of relevant test cases and the sequence in which they are executed in order to perform a conformance test for a given use case.

7.3.1 Test configurations

The test configuration reflects various ISO 15118 scenarios. Entities for the SUT, Tester and PTCs are:

- Electric Vehicle Communication Controller (EVCC);
- Supply Equipment Communication Controller (SECC); and
- PWM Signaling (IEC 61851).

Table 3 shows combinations of such entities defining valid test configurations for this document (CF_Part_ID).

Table 3 — ISO 15118-4 test configurations

CF_Part_ID	SUT	Tester	PTCs
CF_04_001	SECC	EVCC	PTC1=61851
CF_04_002	EVCC	SECC	PTC1=61851

7.3.2 Components and ports

In correspondence to the identified set of relevant test configurations this subclause defines test components which reflect the main entities needed for stimulation of the SUT with respect to ISO 15118. Ports are used to connect these components with each other and the SUT. The components are predefined in the configuration module and are summarized in Table 4. Port types define which kind of messages can be sent or received by this port. The corresponding port type definitions are summarized in Table 5.

Table 4 — Component definitions

Components	Description
SECC_Tester (MTC)	This component type is the main type for the tests of an SECC. A V2G_TCP_TLS_ALM_SECC_Port , V2G_UDP_SDP_Port, HAL_61851_Port and a HAL_61851_Internal_Port (see Table 5) are assigned to this component type.
EVCC_Tester (MTC)	This component type is the main type for the tests of an EVCC. A V2G_TCP_TLS_ALM_EVCC_Port , V2G_UDP_SDP_Port, HAL_61851_Port and a HAL_61851_Internal_Port (see Table 5) are assigned to this component type.
HAL_61851_Listener (PTC)	The HAL_61851_Listener is responsible for watching the correct IEC 61851 behavior and is used in the form of a parallel test component (PTC). This component uses the HAL_61851_Port (see Table 5) to communicate with an IEC 61851-1 SUT Adapter and the HAL_61851_Internal_Port to communicate with the MTC.

Table 5 — Port type definitions

Port Type	Description
V2G_TCP_TLS_ALM_SECC_Port	This port is used to send and receive V2G- Application Layer Messages (ALM) or supportedAppProtocol messages via TCP/TLS defined in ISO 15118-2 to/from an SECC. This port is also used to send and receive internal TCP/TLS port control messages defined in Annex O.
V2G_TCP_TLS_ALM_EVCC_Port	This port is used to send and receive V2G- Application Layer Messages (ALM) or supportedAppProtocol messages via TCP/TLS defined in ISO 15118-2 to/from an EVCC. This port is also used to send and receive internal TCP/TLS port control messages defined in Annex O.
V2G_UDP_SDP_Port	This port is used to send and receive SDP messages via UDP defined in ISO 15118-2 to/from an EVCC or SECC.
HAL_61851_Port	This port is used to send or receive messages to an IEC 61851 communication unit. The communication unit will interpret these messages and will set the corresponding signaling to the SUT (SECC/EVCC). The specification of the IEC 61851 communication unit is out of scope of this document.
HAL_61851_Internal_Port	This port is used to synchronize the status of the IEC 61851-1 PWM signal between the MTC (SECC_Tester or EVCC_Tester) and PTC (HAL_61851_Listener_Port). It enables the MTC and PTC to send and receive

HAL 61851-1 messages.

These components and ports compose relevant test configurations for this document. Whether the type EVCC_Tester or SECC_Tester shall be used as MTC depends on the type of the SUT. If the SUT is an EVCC, the MTC shall use the type EVCC_Tester. If the SUT is an SECC, the MTC shall use the type SECC_Tester. The HAL_61851_Listener combines all necessary listener functionality into one component, independent on whether the MTC represents an SECC_Tester or an EVCC_Tester. The MTC always contains a TTCN-3 test configuration and delimits the lifeline during test execution. Next to using ports for communication purposes, local timers, variables or constants may be assigned to components to store dynamic information during test case execution.

In addition to the MTC and PTC and their corresponding port type definitions, a test configuration also consists of respective Test System Interfaces (TSI). An abstract Test System Interface (TSI) is specified as a collection of ports. A TSI has no local timers, constants or variables. Only ports are assigned to it. During the test case execution test components ports can be mapped dynamically to the TSI ports to establish communication channel to the real test system interface. In the test configuration the TSI uses the type System_EVCC or System_SECC depending on the type of the SUT. If the SUT is an EVCC, the TSI uses the type System_EVCC. If the SUT is an SECC, the TSI uses the type System_SECC.

The test configuration is illustrated in Figure 3. The type of the V2G components and ports (EVCC or SECC) depends on the SUT type.

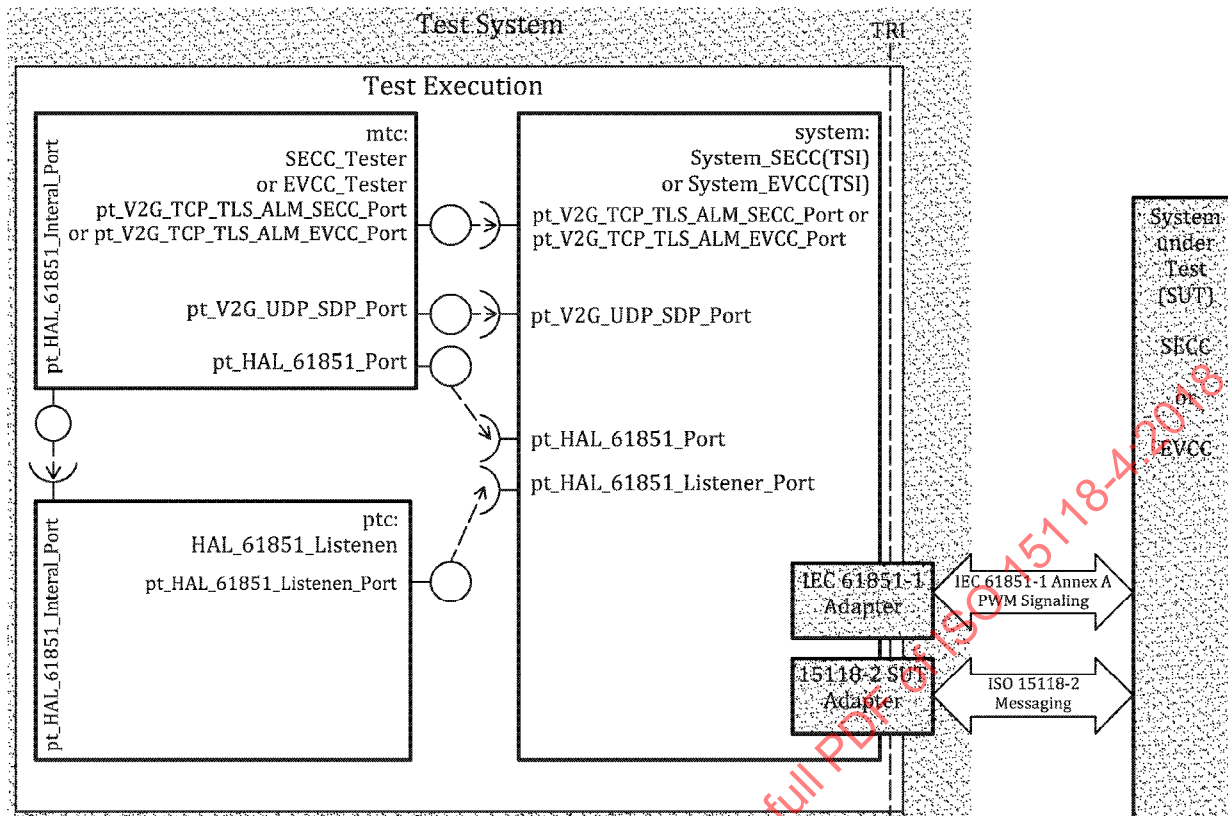


Figure 3 — ISO 15118-4 test configuration

As shown in Figure 3, the port mappings are defined statically as follows.

- The port `pt_V2G_TCP_TLS_ALM_EVCC_Port` of the TSI is always mapped to port `pt_V2G_TCP_TLS_ALM_EVCC_Port` of the MTC.
- The port `pt_V2G_TCP_TLS_ALM_SECC_Port` of the TSI is always mapped to port `pt_V2G_TCP_TLS_ALM_SECC_Port` of the MTC.
- The port `pt_V2G_UDP_SDP_Port` of the TSI is always mapped to port `pt_V2G_UDP_SDP_Port` of the MTC.
- The port `pt_HAL_61851_Port` of the TSI is always mapped to port `pt_HAL_61851_Port` of the MTC.
- The port `pt_HAL_61851_Listener_Port` of the TSI is always mapped to port `pt_HAL_61851_Listener_Port` of the `v_HAL_61851_Listener`. The `pt_HAL_61851_Listener_Port` is from the same type `HAL_61851_Port` as the `pt_HAL_61851_Port`.
- The internal port `pt_HAL_61851_Internal_Port` is not mapped to the TSI. This port is connected between the MTC and PTC.

7.3.3 Protocol Implementation Conformance Statement (PICS) Definition

To evaluate the conformance of a particular implementation, it is necessary to have a statement of the capabilities and options which have been implemented, and any features which have been omitted, so that the implementation can be tested for conformance against relevant requirements, and against those requirements only. Such a statement is called a Protocol Implementation Conformance Statement (PICS) cf. [ITU-T X.290].

All PICS defined in the ATS are summarized in Table 6 - Table 8.

**Table 6 — PICS for test system configurations CF_04_001 and CF_04_002
(SUT either SECC or EVCC)**

PICS	Description
PICS_CMN_CMN_CombinedTesting	Indication for enabling combined testing including SLAC association & V2G messaging
PICS_CMN_CMN_ChargingMode	Indication for testing either AC or DC charging mode
PICS_CMN_CMN_IdentificationMode	Indication for testing either EIM or PnC identification mode
PICS_CMN_CMN_PlugType	Indication for testing either type1 or type2 plug type
PICS_CMN_AC_CableCapability	Indication for used AC cable type Choice: i) capability13A, ii) capability20A, iii) capability32A, iv) capability63A, v) capability70A
PICS_CMN_CMN_WakeUp	Indication for the sleep time within a paused session
PICS_CMN_CMN_V2gtpSdp	Indication for explicit testing V2GTP by using SDP payload
PICS_CMN_CMN_V2gtpV2g	Indication for explicit testing V2GTP by using V2G payload
PICS_CMN_CMN_Renegotiation	Indication for explicit testing of Renegotiation
PICS_CMN_CMN_Renegotiation LoopIndication	Indicates the iteration of the ChargingStatus (AC) or CurrentDemand (DC) loop, when ReNegotiation shall be triggered (shall be smaller or equal to PICS_CMN_CMN_LoopCounter).
PICS_CMN_CMN_LoopCounter	Indicates the number of iterations of the ChargingStatus (AC) or CurrentDemand (DC) loop
PICS_CMN_CMN_InvalidV2gtp DataFields	Indication for enabling Test Group: Invalid V2GTP data fields
PICS_CMN_CMN_SDPTimeouts AndInvalidData	Indication for enabling Test Group: SDP Timeouts and invalid SDP data fields
PICS_CMN_CMN_FailedResponseCodes	Indication for enabling Test Group: V2G FAILED response codes
PICS_CMN_CMN_V2gTimeouts	Indication for enabling Test Group: V2G timeouts
PICS_CMN_CMN_PkiName	Indication for selected PKI Choice: i) PKI_1, ii) PKI_2, iii) PKI_3, iv) PKI_4, v) PKI_5, vi) PKI_6, vii) PKI_7

**Table 7 — PICS for test system configurations CF_04_001
(SUT equals SECC)**

PICS	Description
PICS_SECC_CMN_Pause	Indication for enabling a charging pause
PICS_SECC_CMN_EIMDone	Indication for initiating EIM Authorization process Choice: i) beforePlugin, ii) afterPlugin, iii) duringSlac, iv) v2gAuthorization
PICS_SECC_CMN_EvStateInChargeProgr ess	Indication for explicit testing of State C/D of IEC 61851-1:2017, Annex A during charging
PICS_SECC_DC_WeldingDetection	Indication for explicit testing of WeldingDetection
PICS_SECC_CMN_DepartureTime	Indication for using a departure time
PICS_SECC_CMN_MaxEntriesSASchedul eTuple	Indication for enabling a maximal number of entries in the SAScheduleTuple

**Table 8 — PICS for test system configurations CF_04_002
(SUT equals EVCC)**

PICS	Description
PICS_EVCC_CMN_PmaxSchedulewithZeroPow	Indication for enabling a PmaxScheduleList with a list element pMax = 0W (triggering pause) Choice: i) sleepWithoutCharge, ii) sleepAfterCharge, iii) none_
PICS_EVCC_CMN_ReceiptRequired	Indication for explicit testing of MeteringReceipt message
PICS_EVCC_CMN_InvalidStatesAndDutyCycles	Indication for enabling Test Group: Invalid states and duty cycles
PICS_EVCC_CMN_InvalidV2GMessages	Indication for enabling Test Group: Invalid V2G messages
PICS_CMN_EVCC_SalesTariff	Indication for using a SalesTariff

7.3.4 Protocol Implementation extra Information for Testing (PIXIT) Definition

In order to ensure testability of each defined message pair defined in ISO 15118-2, the following set of PIXIT are defined in addition to PICS in this document. PIXIT are used to indicate to the SUT that a specific capability of the SUT is tested.

NOTE 1 Due to the black box test paradigm in this document, it is not defined how to ensure that a corresponding PIXIT is set on the SUT side for a given test case execution.

All PIXIT defined in the ATS are summarized in Table 9 - Table 11.

**Table 9 — Selected PIXIT for test system configurations CF_04_001 and CF_04_002
(SUT either SECC or EVCC)**

PIXIT	Description
PIXIT_CMN_CMN_WakeUp	Indication for the wake-up time (SUT) within a paused session
PIXIT_CMN_CMN_CertificateHandling	Indication for explicit testing of CertificateInstallation or CertificateUpdate message
PIXIT_CMN_CMN_PkiName	Indication for selected PKI Choice: i) PKI_1, ii) PKI_2, iii) PKI_3, iv) PKI_4, v) PKI_5, vi) PKI_6, vii) PKI_7
PIXIT_CMN_CMN_Renegotiation	Indication for explicit testing of Renegotiation initiated by the SUT Choice: i) renegotiation, ii) unknown
PIXIT_CMN_CMN_MeterReading	Indication for explicit testing of MeterReading element Choice: i) meterReading, ii) unknown

**Table 10 — Selected PIXIT for test system configurations CF_04_001
(SUT equals SECC)**

PIXIT	Description
PIXIT_SECC_CMN_VAS	Indication for explicit testing of ServiceDetail message (SUT shall send a serviceList within ServiceDiscoveryRes message) Choice: i) serviceDetail, ii) none_, iii) unknown
PIXIT_SECC_CMN_ReceiptRequired	Indication for explicit testing of MeteringReceipt message
PIXIT_SECC_CMN_CertificateHandling	Indication for explicit testing of CertificateInstallation or CertificateUpdate message

PIXIT	Description
	Choice: i) certInstall, ii) certUpdate, iii) none_
PIXIT_SECC_CMN_SalesTariff	Indication for explicit testing of Sales Tariff elements Choice: i) salesTariff, ii) unknown
PIXIT_SECC_CMN_TLS	Indication for explicit testing of TLS in EIM mode
PIXIT_SECC_CMN_VersionNumberMinor	Indication for the supported VersionNumberMinor of the SUT
PIXIT_SECC_CMN_InternetAccess	Indication for support of InternetAccess service Choice: i) internetAccess, ii) unknown
PIXIT_SECC_DC_CableCheckOngoing	Indication for explicit testing of the repetition of the CableCheck message sequence Choice: i) ongoing, ii) unknown
PIXIT_SECC_CMN_AuthorizationOngoing	Indication for explicit testing of the repetition of the Authorization message sequence until V2G_SECC_Ongoing_Timer expires on SUT side Choice: i) ongoing, ii) unknown
PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing	Indication for explicit testing of the repetition of the ChargeParameterDiscovery message sequence until V2G_SECC_Ongoing_Timer expires on SUT side Choice: i) ongoing, ii) unknown
PIXIT_SECC_CMN_MeterInfo	Indication for explicit testing of MeterInfo element Choice: i) meterInfo, ii) unknown
PIXIT_SECC_CMN_PaymentOption	Indication for supported payment option within ServiceDiscoveryRes message (Only one payment option possible) Choice: i) externalPayment, ii) contract

**Table 11 — Selected PIXIT for test system configurations CF_04_002
(SUT equals EVCC)**

PIXIT	Description
PIXIT_EVCC_CMN_VAS	Indication for explicit testing of ServiceDetail message (SUT shall send a ServiceDetailReq message for each service of the received serviceList) Choice: i) serviceDetail, ii) none_, iii) unknown
PIXIT_EVCC_CMN_Pause	Indication for enabling a charging pause Choice: i) pause, ii) unknown, iii) none_
PIXIT_EVCC_DC_WeldingDetection	Indication for explicit testing of WeldingDetection message Choice: i) weldingDetection, ii) none_, iii) unknown
PIXIT_EVCC_CMN_TLS	Indication for explicit testing of TLS in EIM mode Choice: i) tls, ii) none_, iii) unknown
PIXIT_EVCC_CMN_CertificateHandling	Indication for explicit testing of CertificateInstallation or CertificateUpdate message Choice: i) certInstall, ii) certUpdate, iii) none_, iv) unknown
PIXIT_EVCC_CMN_DepartureTime	Indication for explicit testing of DepartureTime element Choice: i) departureTime, ii) none_, iii) unknown
PIXIT_EVCC_AC_MaxNumberOfPhasesInUse	Indication for explicit testing of MaxNumberOfPhasesInUse element Choice: i) maxNumberOfPhasesInUse, ii) unknown

PIXIT	Description
PIXIT_EVCC_CMN_ChargingProfileOptimization	Indication for the SUTs capability to optimize charging profiles based on the charge schedule Choice: i) optimization, ii) none, iii) unknown
PIXIT_EVCC_CMN_NewSAScheduleTupleID	Indication for explicit testing of using a new SAScheduleTupleID after renegotiation by the SUT Choice: i) newSAScheduleTupleID, ii) unknown
PIXIT_EVCC_CMN_SAScheduleTupleID MR	Indication for explicit testing SAScheduleTupleID within MeteringReceiptReq message Choice: i) sAScheduleTupleID, ii) unknown

7.3.5 Test control

The control part calls the test cases with actual parameters and controls their execution. Program statements PICS/PIXIT are used to specify the selection and execution order of the test cases (see Annex B). For each use case all valid PICS/PIXIT configurations are summarized in Table 12 - Table 15.

[V2G4-018] The test control for testing an AC SECC shall execute all common and all AC SECC specific test cases for each PICS/PIXIT configuration, as defined in Table 12.

Table 12 — SECC AC PICS/PIXIT configuration

Configuration	EIM	PnC
1	PICS_CMN_CMN_V2gtpSdp := true and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_V2gtpSdp := true and PICS_CMN_CMN_ChargingMode := aC
2	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_TLS := false and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_ChargingMode := aC
3	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_TLS := true and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_V2gtpV2g := true and PICS_CMN_CMN_ChargingMode := aC
4	PICS_CMN_CMN_V2gtpV2g := true and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
5	PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
6	PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_SECC_CMN_InternetAccess := unknown and PICS_CMN_CMN_ChargingMode := aC
7	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_VAS := serviceDetail and PIXIT_SECC_CMN_TLS := true and PIXIT_SECC_CMN_InternetAccess := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_SECC_CMN_InternetAccess := internetAccess and PICS_CMN_CMN_ChargingMode := aC
8	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_VAS := serviceDetail and PIXIT_SECC_CMN_TLS := true and	PIXIT_SECC_CMN_VAS := none_ and PICS_CMN_CMN_ChargingMode := aC

Configuration	EIM	PnC
	PIXIT_SECC_CMN_InternetAccess := internetAccess and PICS_CMN_CMN_ChargingMode := aC	
9	PIXIT_SECC_CMN_VAS := none and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC
10	PICS_CMN_CMN_IdentificationMode := eIM and PICS_CMN_CMN_ChargingMode := aC	PIXIT_SECC_CMN_CertificateHandling := certInstall and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC
11	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := aC	PIXIT_SECC_CMN_CertificateHandling := certUpdate and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC
12	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := aC	PIXIT_SECC_CMN_CertificateHandling := none and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC
13	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := aC
14	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := aC
15	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_ChargingMode := aC
16	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := aC
17	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_ChargingMode := aC
18	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_ChargingMode := aC

Configuration	EIM	PnC
	PICS_CMN_CMN_ChargingMode := aC	
19	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := aC
20	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
21	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
22	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
23	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
24	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC

Configuration	EIM	PnC
25	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
26	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
27	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
28	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
29	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
30	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC

Configuration	EIM	PnC
31	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
32	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
33	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
34	PICS_SECC_CMN_EvStateInChargeProgress := C and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
35	PICS_SECC_CMN_EvStateInChargeProgress := D and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
36	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_MeterInfo := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_EvStateInChargeProgress := C and PICS_CMN_CMN_ChargingMode := aC
37	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_MeterInfo := meterInfo and PIXIT_CMN_CMN_MeterReading := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_EvStateInChargeProgress := D and PICS_CMN_CMN_ChargingMode := aC

Configuration	EIM	PnC
38	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_MeterInfo := meterInfo and PIXIT_CMN_CMN_MeterReading := meterReading and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_ReceiptRequired := false and PIXIT_SECC_CMN_MeterInfo := unknown and PICS_CMN_CMN_ChargingMode := aC
39	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_ReceiptRequired := false and PIXIT_SECC_CMN_MeterInfo := meterInfo and PIXIT_CMN_CMN_MeterReading := unknown and PICS_CMN_CMN_ChargingMode := aC
40	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_ReceiptRequired := false and PIXIT_SECC_CMN_MeterInfo := meterInfo and PIXIT_CMN_CMN_MeterReading := meterReading and PICS_CMN_CMN_ChargingMode := aC
41	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_ReceiptRequired := true and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_SECC_CMN_MeterInfo := unknown and PICS_CMN_CMN_ChargingMode := aC
42	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_ReceiptRequired := true and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_SECC_CMN_MeterInfo := meterInfo and PIXIT_CMN_CMN_MeterReading := unknown and PICS_CMN_CMN_ChargingMode := aC
43	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_ReceiptRequired := true and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_SECC_CMN_MeterInfo := meterInfo and PIXIT_CMN_CMN_MeterReading := meterReading and PICS_CMN_CMN_ChargingMode := aC
44	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC
45	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC

Configuration	EIM	PnC
46	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC
47	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC
48	PIXIT_SECC_CMN_VAS := serviceDetail and PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_TLS := true and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC
49	PICS_CMN_CMN_IdentificationMode := eIM and PICS_SECC_CMN_EIMDone := v2gAuthorization and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC
50	PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing := ongoing and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC
51	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_PaymentOption := externalPayment and PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC
52	PICS_CMN_CMN_ChargingMode := aC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := aC
53		PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_ReceiptRequired := false and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC
54		PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_AuthorizationOngoing := ongoing and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC

Configuration	EIM	PnC
55		PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_CertificateHandling := certInstall and PIXIT_SECC_CMN_AuthorizationOngoing := ongoing and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC
56		PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_CertificateHandling := certUpdate and PIXIT_SECC_CMN_AuthorizationOngoing := ongoing and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC
57		PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing := ongoing and PICS_CMN_CMN_ChargingMode := aC
58		PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_PaymentOption := contract and PICS_CMN_CMN_ChargingMode := aC
59		PICS_CMN_CMN_PkiType := PKI-1 and PIXIT_CMN_CMN_PkiType := PKI-1 and PICS_CMN_CMN_ChargingMode := aC
60		PICS_CMN_CMN_PkiType := PKI-2 and PIXIT_CMN_CMN_PkiType := PKI-2 and PICS_CMN_CMN_ChargingMode := aC
61		PICS_CMN_CMN_PkiType := PKI-3 and PIXIT_CMN_CMN_PkiType := PKI-3 and PICS_CMN_CMN_ChargingMode := aC
62		PICS_CMN_CMN_PkiType := PKI-4 and PIXIT_CMN_CMN_PkiType := PKI-4 and PICS_CMN_CMN_ChargingMode := aC
63		PICS_CMN_CMN_PkiType := PKI-5 and PIXIT_CMN_CMN_PkiType := PKI-5 and PICS_CMN_CMN_ChargingMode := aC
64		PICS_CMN_CMN_PkiType := PKI-6 and PIXIT_CMN_CMN_PkiType := PKI-6 and PICS_CMN_CMN_ChargingMode := aC
65		PICS_CMN_CMN_PkiType := PKI-7 and PIXIT_CMN_CMN_PkiType := PKI-7 and PICS_CMN_CMN_ChargingMode := aC
66		PICS_CMN_CMN_ChargingMode := aC

[V2G4-019] The test control for testing a DC SECC shall execute all common and all DC SECC specific test cases for each PICS/PIXIT configuration, as defined in Table 13.

Table 13 — SECC DC PICS/PIXIT configuration

Configuration	EIM	PnC
1	PICS_CMN_CMN_V2gtpSdp := true and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_V2gtpSdp := true and PICS_CMN_CMN_ChargingMode := dC
2	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_TLS := false and	PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_ChargingMode := dC

Configuration	EIM	PnC
	PICS_CMN_CMN_ChargingMode := dC	
3	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_TLS := true and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_V2gtpV2g := true and PICS_CMN_CMN_ChargingMode := dC
4	PICS_CMN_CMN_V2gtpV2g := true and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
5	PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
6	PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_SECC_CMN_InternetAccess := unknown and PICS_CMN_CMN_ChargingMode := dC
7	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_VAS := serviceDetail and PIXIT_SECC_CMN_TLS := true and PIXIT_SECC_CMN_InternetAccess := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_SECC_CMN_InternetAccess := internetAccess and PICS_CMN_CMN_ChargingMode := dC
8	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_VAS := serviceDetail and PIXIT_SECC_CMN_TLS := true and PIXIT_SECC_CMN_InternetAccess := internetAccess and PICS_CMN_CMN_ChargingMode := dC	PIXIT_SECC_CMN_VAS := none_ and PICS_CMN_CMN_ChargingMode := dC
9	PIXIT_SECC_CMN_VAS := none_ and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC
10	PICS_CMN_CMN_IdentificationMode := eIM and PICS_CMN_CMN_ChargingMode := dC	PIXIT_SECC_CMN_CertificateHandling := certInstall and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC

Configuration	EIM	PnC
11	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := dC	PIXIT_SECC_CMN_CertificateHandling := certUpdate and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC
12	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := dC	PIXIT_SECC_CMN_CertificateHandling := none_ and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC
13	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := dC
14	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := dC
15	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_ChargingMode := dC
16	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := dC
17	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_ChargingMode := dC
18	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_ChargingMode := dC
19	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_ChargingMode := dC
20	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and

[illegible]

Configuration	EIM	PnC
	PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
28	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
29	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
30	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
31	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
32	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
33	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
34	PICS_SECC_CMN_EvStateInChargeProgress := C and PIXIT_SECC_DC_CableCheckOngoing := unknown	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and

Configuration	EIM	PnC
	and PICS_CMN_CMN_ChargingMode := dC	PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
35	PICS_SECC_CMN_EvStateInChargeProgress := D and PIXIT_SECC_DC_CableCheckOngoing := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_SECC_CMN_Pause := true and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
36	PICS_SECC_CMN_EvStateInChargeProgress := C and PIXIT_SECC_DC_CableCheckOngoing := ongoing and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_EvStateInChargeProgress := C and PIXIT_SECC_DC_CableCheckOngoing := ongoing and PICS_CMN_CMN_ChargingMode := dC
37	PICS_SECC_CMN_EvStateInChargeProgress := D and PIXIT_SECC_DC_CableCheckOngoing := ongoing and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_EvStateInChargeProgress := D and PIXIT_SECC_DC_CableCheckOngoing := ongoing and PICS_CMN_CMN_ChargingMode := dC
38	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_MeterInfo := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_EvStateInChargeProgress := C and PIXIT_SECC_DC_CableCheckOngoing := ongoing and PICS_CMN_CMN_ChargingMode := dC
39	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_MeterInfo := meterInfo and PIXIT_CMN_CMN_MeterReading := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_EvStateInChargeProgress := D and PIXIT_SECC_DC_CableCheckOngoing := ongoing and PICS_CMN_CMN_ChargingMode := dC
40	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_MeterInfo := meterInfo and PIXIT_CMN_CMN_MeterReading := meterReading and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_ReceiptRequired := false and PIXIT_SECC_CMN_MeterInfo := unknown and PICS_CMN_CMN_ChargingMode := dC
41	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_ReceiptRequired := false and PIXIT_SECC_CMN_MeterInfo := meterInfo and PIXIT_CMN_CMN_MeterReading := unknown and PICS_CMN_CMN_ChargingMode := dC
42	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_ReceiptRequired := false and PIXIT_SECC_CMN_MeterInfo := meterInfo and PIXIT_CMN_CMN_MeterReading := meterReading and PICS_CMN_CMN_ChargingMode := dC
43	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_ReceiptRequired := true and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_SECC_CMN_MeterInfo := unknown and PICS_CMN_CMN_ChargingMode := dC

Configuration	EIM	PnC
44	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_ReceiptRequired := true and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_SECC_CMN_MeterInfo := meterInfo and PIXIT_CMN_CMN_MeterReading := unknown and PICS_CMN_CMN_ChargingMode := dC
45	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_ReceiptRequired := true and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_SECC_CMN_MeterInfo := meterInfo and PIXIT_CMN_CMN_MeterReading := meterReading and PICS_CMN_CMN_ChargingMode := dC
46	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC
47	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC
48	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC
49	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC
50	PICS_SECC_DC_WeldingDetection := true and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC
51	PICS_SECC_DC_WeldingDetection := false and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC
52	PIXIT_SECC_CMN_VAS := serviceDetail and PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_TLS := true and	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := false and PIXIT_SECC_CMN_SalesTariff := salesTariff and

Configuration	EIM	PnC
	PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC
53	PICS_CMN_CMN_IdentificationMode := eIM and PICS_SECC_CMN_EIMDone := v2gAuthorization and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := salesTariff and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC
54	PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing := ongoing and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true and PICS_SECC_CMN_DepartureTime := true and PIXIT_SECC_CMN_SalesTariff := unknown and PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_ChargingMode := dC
55	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_SECC_CMN_PaymentOption := externalPayment and PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_DC_WeldingDetection := true and PICS_CMN_CMN_ChargingMode := dC
56	PICS_CMN_CMN_ChargingMode := dC	PICS_SECC_DC_WeldingDetection := false and PICS_CMN_CMN_ChargingMode := dC
57		PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_ReceiptRequired := true and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC
58		PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_AuthorizationOngoing := ongoing and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC
59		PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_CertificateHandling := certInstall and PIXIT_SECC_CMN_AuthorizationOngoing := ongoing and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC
60		PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_CertificateHandling := certUpdate and PIXIT_SECC_CMN_AuthorizationOngoing := ongoing and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC
61		PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing := ongoing and PICS_CMN_CMN_ChargingMode := dC
62		PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_SECC_CMN_PaymentOption := contract and PICS_CMN_CMN_ChargingMode := dC
63		PICS_CMN_CMN_PkiType := PKI-1 and PIXIT_CMN_CMN_PkiType := PKI-1 and PICS_CMN_CMN_ChargingMode := dC

Configuration	EIM	PnC
64		PICS_CMN_CMN_PkiType := PKI-2 and PIXIT_CMN_CMN_PkiType := PKI-2 and PICS_CMN_CMN_ChargingMode := dC
65		PICS_CMN_CMN_PkiType := PKI-3 and PIXIT_CMN_CMN_PkiType := PKI-3 and PICS_CMN_CMN_ChargingMode := dC
66		PICS_CMN_CMN_PkiType := PKI-4 and PIXIT_CMN_CMN_PkiType := PKI-4 and PICS_CMN_CMN_ChargingMode := dC
67		PICS_CMN_CMN_PkiType := PKI-5 and PIXIT_CMN_CMN_PkiType := PKI-5 and PICS_CMN_CMN_ChargingMode := dC
68		PICS_CMN_CMN_PkiType := PKI-6 and PIXIT_CMN_CMN_PkiType := PKI-6 and PICS_CMN_CMN_ChargingMode := dC
69		PICS_CMN_CMN_PkiType := PKI-7 and PIXIT_CMN_CMN_PkiType := PKI-7 and PICS_CMN_CMN_ChargingMode := dC
70		PICS_CMN_CMN_ChargingMode := dC

[V2G4-020] The test control for testing an AC EVCC shall execute all common and all AC EVCC specific test cases for each PICS/PIXIT configuration, as defined in Table 14.

Table 14 — EVCC AC PICS/PIXIT configuration

Configuration	EIM	PnC
1	PICS_CMN_CMN_V2gtpSdp := true and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_V2gtpSdp := true and PICS_CMN_CMN_ChargingMode := aC
2	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_ChargingMode := aC
3	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tls and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
4	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC

Configuration	EIM	PnC
5	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
6	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
7	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_V2gtpV2g := true and PICS_CMN_CMN_ChargingMode := aC
8	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
9	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
10	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
11	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC

Configuration	EIM	PnC
12	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tls and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC
13	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := none and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC
14	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := unknown and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC
15	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tls and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := aC
16	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := none and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := aC
17	PICS_CMN_CMN_V2gtpV2g := true and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := unknown and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := aC

Configuration	EIM	PnC
18	PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := aC
19	PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := none_ and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := aC
20	PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := unknown and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := aC
21	PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := aC
22	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := serviceDetail and PIXIT_EVCC_CMN_TLS := tls and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := none_ and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := aC
23	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := none_ and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := unknown and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := aC
24	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp

Configuration	EIM	PnC
		PICS_CMN_CMN_ChargingMode := aC
25	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := serviceDetail and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := none_ and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := aC
26	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := unknown and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := aC
27	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_CertificateHandling := certInstall and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC
28	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := serviceDetail and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_CertificateHandling := certUpdate and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC
29	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_CertificateHandling := none_ and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC
30	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_CertificateHandling := unknown and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC
31	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := serviceDetail and PIXIT_EVCC_CMN_TLS := tIs and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_CMN_CMN_ChargingMode := aC
32	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and	PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_CMN_CMN_ChargingMode := aC

Configuration	EIM	PnC
	PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	
33	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and Ppar_SECC_Pmax0W IXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_CMN_DepartureTime := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
34	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := serviceDetail and PIXIT_EVCC_CMN_TLS := tIs and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_CMN_DepartureTime := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
35	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_CMN_DepartureTime := departureTime and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
36	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and Ppar_SECC_Pmax0W IXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_CMN_DepartureTime := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
37	PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_CMN_DepartureTime := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
38	PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_CMN_DepartureTime := departureTime and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
39	PIXIT_EVCC_CMN_DepartureTime := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
40	PIXIT_EVCC_CMN_DepartureTime := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and	PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and

Configuration	EIM	PnC
	PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
41	PIXIT_EVCC_CMN_DepartureTime := departureTime and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_CMN_DepartureTime := departureTime and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
42	PIXIT_EVCC_CMN_DepartureTime := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
43	PIXIT_EVCC_CMN_DepartureTime := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
44	PIXIT_EVCC_CMN_DepartureTime := departureTime and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_CMN_DepartureTime := departureTime and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
45	PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PICS_CMN_CMN_ChargingMode := aC
46	PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PICS_CMN_CMN_ChargingMode := aC
47	PIXIT_EVCC_CMN_DepartureTime := departureTime and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_CMN_ChargingMode := aC
48	PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow :=	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization :=

Configuration	EIM	PnC
	sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	optimization and PICS_CMN_CMN_ChargingMode := aC
49	PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_CMN_ChargingMode := aC
50	PIXIT_EVCC_CMN_DepartureTime := departureTime and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_CMN_ChargingMode := aC
51	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_MeterReading := unknown and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown and PICS_CMN_CMN_ChargingMode := aC
52	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling := certUpdate and PIXIT_CMN_CMN_MeterReading := unknown and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown and PICS_CMN_CMN_ChargingMode := aC
53	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_CMN_ChargingMode := aC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling := certInstall and PIXIT_CMN_CMN_MeterReading := unknown and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown and PICS_CMN_CMN_ChargingMode := aC
54	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_CMN_ChargingMode := aC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_MeterReading := meterReading and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown and PICS_CMN_CMN_ChargingMode := aC
55	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling :=

Configuration	EIM	PnC
	PICS_CMN_CMN_ChargingMode := aC	certUpdate and PIXIT_CMN_CMN_MeterReading := meterReading and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown and PICS_CMN_CMN_ChargingMode := aC
56	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := none and PICS_CMN_CMN_ChargingMode := aC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling := certInstall and PIXIT_CMN_CMN_MeterReading := meterReading and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown and PICS_CMN_CMN_ChargingMode := aC
57	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_IdentificationMode := eIM and PICS_CMN_CMN_ChargingMode := aC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_MeterReading := unknown and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID and PICS_CMN_CMN_ChargingMode := aC
58	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PICS_CMN_CMN_IdentificationMode := eIM and PICS_CMN_CMN_ChargingMode := aC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling := certUpdate and PIXIT_CMN_CMN_MeterReading := unknown and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID and PICS_CMN_CMN_ChargingMode := aC
59	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling := certInstall and PIXIT_CMN_CMN_MeterReading := unknown and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID and PICS_CMN_CMN_ChargingMode := aC
60	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_MeterReading := meterReading and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID and PICS_CMN_CMN_ChargingMode := aC

Configuration	EIM	PnC
61	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_CMN_CMN_ChargingMode := aC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling := certUpdate and PIXIT_CMN_CMN_MeterReading := meterReading and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID and PICS_CMN_CMN_ChargingMode := aC
62	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_CMN_CMN_ChargingMode := aC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling := certInstall and PIXIT_CMN_CMN_MeterReading := meterReading and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID and PICS_CMN_CMN_ChargingMode := aC
63	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := aC
64	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_IdentificationMode := pnC and PICS_EVCC_CMN_ReceiptRequired := false and PICS_CMN_CMN_ChargingMode := aC
65	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PICS_CMN_CMN_IdentificationMode := pnC and PICS_EVCC_CMN_ReceiptRequired := false and PICS_CMN_CMN_ChargingMode := aC
66	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_ChargingMode := aC
67	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation

Configuration	EIM	PnC
	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC	and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_ChargingMode := aC
68	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC
69	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC
70	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC
71	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC
72	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC
73	PICS_CMN_CMN_Renegotiation := true and	PICS_CMN_CMN_Renegotiation := false and

Configuration	EIM	PnC
	PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC	PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC
74	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC
75	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID := newSAScheduleTupleID and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC
76	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := newSAScheduleTupleID and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC
77	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := newSAScheduleTupleID and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC
78	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID :=	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and

Configuration	EIM	PnC
	newSAScheduleTupleID and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_ChargingMode := aC
79	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := aC
80	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID := newSAScheduleTupleID and PICS_CMN_CMN_ChargingMode := aC
81	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := newSAScheduleTupleID and PICS_CMN_CMN_ChargingMode := aC
82	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_CMN_Pause := none_ and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := newSAScheduleTupleID and PICS_CMN_CMN_ChargingMode := aC
83	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_CMN_Pause := none_ and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := newSAScheduleTupleID and PICS_CMN_CMN_ChargingMode := aC
84	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_CMN_Pause := none_ and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := aC
85	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_CMN_Pause := pause and	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_Pause := none_ and PICS_CMN_CMN_ChargingMode := aC

Configuration	EIM	PnC
	PICS_CMN_CMN_ChargingMode := aC	
86	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_ChargingMode := aC
87	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := aC
88	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_IdentificationMode := pnC and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := aC
89	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_PkiType := PKI-1 and PIXIT_CMN_CMN_PkiType := PKI-1 and PICS_CMN_CMN_ChargingMode := aC
90	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_PkiType := PKI-2 and PIXIT_CMN_CMN_PkiType := PKI-2 and PICS_CMN_CMN_ChargingMode := aC
91	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_PkiType := PKI-3 and PIXIT_CMN_CMN_PkiType := PKI-3 and PICS_CMN_CMN_ChargingMode := aC
92	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_PkiType := PKI-4 and PIXIT_CMN_CMN_PkiType := PKI-4 and PICS_CMN_CMN_ChargingMode := aC
93	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := aC	PICS_CMN_CMN_PkiType := PKI-5 and PIXIT_CMN_CMN_PkiType := PKI-5 and PICS_CMN_CMN_ChargingMode := aC
94		PICS_CMN_CMN_PkiType := PKI-6 and PIXIT_CMN_CMN_PkiType := PKI-6 and PICS_CMN_CMN_ChargingMode := aC
95		PICS_CMN_CMN_PkiType := PKI-7 and PIXIT_CMN_CMN_PkiType := PKI-7 and PICS_CMN_CMN_ChargingMode := aC
96		PICS_CMN_CMN_ChargingMode := aC

[V2G4-021] The test control for testing a DC EVCC shall execute all common and all DC EVCC specific test cases for each PICS/PIXIT configuration, as defined in Table 15.

Table 15 — EVCC DC PICS/PIXIT configuration

Configuration	EIM	PnC
1	PICS_CMN_CMN_V2gtpSdp := true and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_V2gtpSdp := true and PICS_CMN_CMN_ChargingMode := dC

Configuration	EIM	PnC
2	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_ChargingMode := dC
3	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
4	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
5	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
6	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
7	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_V2gtpV2g := true and PICS_CMN_CMN_ChargingMode := dC
8	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
9	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
10	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
11	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none and	PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and

Configuration	EIM	PnC
	PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
12	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tls and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC
13	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := none and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC
14	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := unknown and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC
15	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tls and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := dC
16	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := none and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := dC
17	PICS_CMN_CMN_V2gtpV2g := true and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := unknown and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := dC
18	PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_Pause := pause and

Configuration	EIM	PnC
	PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := dC
19	PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := none and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := dC
20	PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := unknown and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := dC
21	PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := dC
22	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := serviceDetail and PIXIT_EVCC_CMN_TLS := tls and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := none and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := dC
23	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := none and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := unknown and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := dC
24	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := serviceDetail and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := dC
25	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := serviceDetail and PIXIT_EVCC_CMN_TLS := tls and PIXIT_EVCC_CMN_Pause := pause and	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := none and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and

Configuration	EIM	PnC
	PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := dC
26	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_VAS := unknown and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp PICS_CMN_CMN_ChargingMode := dC
27	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_CertificateHandling := certInstall and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC
28	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := serviceDetail and PIXIT_EVCC_CMN_TLS := tls and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_CertificateHandling := certUpdate and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC
29	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_CertificateHandling := none_ and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC
30	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_CMN_CertificateHandling := unknown and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC
31	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := serviceDetail and PIXIT_EVCC_CMN_TLS := tls and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_CMN_CMN_ChargingMode := dC
32	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_CMN_CMN_ChargingMode := dC
33	PICS_CMN_CMN_IdentificationMode := eIM and	PIXIT_EVCC_CMN_DepartureTime := unknown

Configuration	EIM	PnC
	PIXIT_EVCC_CMN_VAS := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and Ppar_SECC_Pmax0W IXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := aC
34	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := serviceDetail and PIXIT_EVCC_CMN_TLS := tIs and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_DepartureTime := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
35	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_DepartureTime := departureTime and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
36	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_VAS := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and Ppar_SECC_Pmax0W IXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_DepartureTime := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
37	PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_DepartureTime := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
38	PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_DepartureTime := departureTime and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
39	PIXIT_EVCC_CMN_DepartureTime := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
40	PIXIT_EVCC_CMN_DepartureTime := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
41	PIXIT_EVCC_CMN_DepartureTime :=	PIXIT_EVCC_CMN_DepartureTime :=

Configuration	EIM	PnC
	departureTime and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	departureTime and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
42	PIXIT_EVCC_CMN_DepartureTime := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
43	PIXIT_EVCC_CMN_DepartureTime := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
44	PIXIT_EVCC_CMN_DepartureTime := departureTime and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_CombinedTesting := true and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_DepartureTime := departureTime and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC
45	PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PICS_CMN_CMN_ChargingMode := dC
46	PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_EVCC_SalesTariff := true and PICS_CMN_CMN_ChargingMode := dC
47	PIXIT_EVCC_CMN_DepartureTime := departureTime and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_EVCC_SalesTariff := true and PICS_CMN_CMN_ChargingMode := dC
48	PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_MeterReading := unknown and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown and PICS_CMN_CMN_ChargingMode := dC

Configuration	EIM	PnC
49	PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling := certUpdate and PIXIT_CMN_CMN_MeterReading := unknown and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown and PICS_CMN_CMN_ChargingMode := dC
50	PIXIT_EVCC_CMN_DepartureTime := departureTime and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_CombinedTesting := true and par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp and PICS_CMN_CMN_ChargingMode := dC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling := certInstall and PIXIT_CMN_CMN_MeterReading := unknown and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown and PICS_CMN_CMN_ChargingMode := dC
51	PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_MeterReading := meterReading and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown and PICS_CMN_CMN_ChargingMode := dC
52	PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_EVCC_SalesTariff := true and PICS_CMN_CMN_ChargingMode := dC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling := certUpdate and PIXIT_CMN_CMN_MeterReading := meterReading and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown and PICS_CMN_CMN_ChargingMode := dC
53	PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_EVCC_SalesTariff := true and PICS_CMN_CMN_ChargingMode := dC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling := certInstall and PIXIT_CMN_CMN_MeterReading := meterReading and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown and PICS_CMN_CMN_ChargingMode := dC
54	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_IdentificationMode := eIM and PICS_CMN_CMN_ChargingMode := dC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_MeterReading := unknown and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID and PICS_CMN_CMN_ChargingMode := dC
55	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and

Configuration	EIM	PnC
	PICS_CMN_CMN_IdentificationMode := eIM and PICS_CMN_CMN_ChargingMode := dC	PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling := certUpdate and PIXIT_CMN_CMN_MeterReading := unknown and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID and PICS_CMN_CMN_ChargingMode := dC
56	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling := certInstall and PIXIT_CMN_CMN_MeterReading := unknown and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID and PICS_CMN_CMN_ChargingMode := dC
57	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_MeterReading := meterReading and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID and PICS_CMN_CMN_ChargingMode := dC
58	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_CMN_DepartureTime := none and PICS_CMN_CMN_ChargingMode := dC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling := certUpdate and PIXIT_CMN_CMN_MeterReading := meterReading and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID and PICS_CMN_CMN_ChargingMode := dC
59	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_CMN_DepartureTime := none and PICS_CMN_CMN_ChargingMode := dC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PIXIT_EVCC_CMN_CertificateHandling := certInstall and PIXIT_CMN_CMN_MeterReading := meterReading and PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID and PICS_CMN_CMN_ChargingMode := dC
60	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_CMN_CMN_ChargingMode := dC
61	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_IdentificationMode := pnC and PICS_EVCC_CMN_ReceiptRequired := false and PICS_CMN_CMN_ChargingMode := dC

Configuration	EIM	PnC
	PICS_CMN_CMN_ChargingMode := dC	
62	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PICS_CMN_CMN_IdentificationMode := pnC and PICS_EVCC_CMN_ReceiptRequired := false and PICS_CMN_CMN_ChargingMode := dC
63	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := unknown and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_ChargingMode := dC
64	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PICS_CMN_CMN_IdentificationMode := pnC and PICS_CMN_CMN_PkiName := PKI_1 and PIXIT_CMN_CMN_PkiName := PKI_1 and PICS_EVCC_CMN_ReceiptRequired := true and PICS_CMN_CMN_ChargingMode := dC
65	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_CMN_CMN_ChargingMode := dC
66	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID := newSAScheduleTupleID and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_CMN_DepartureTime := unknown and PICS_CMN_CMN_ChargingMode := dC
67	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := newSAScheduleTupleID and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_CMN_CMN_ChargingMode := dC
68	PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_CMN_DepartureTime := none_ and PICS_CMN_CMN_ChargingMode := dC
69	PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID :=

Configuration	EIM	PnC
		unknown and PICS_CMN_CMN_ChargingMode := dC
70	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := dC
71	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_DC_WeldingDetection := none_ and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := dC
72	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_DC_WeldingDetection := unknown and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := dC
73	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := dC
74	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_DC_WeldingDetection := none_ and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_CMN_ChargingProfileOptimization := none_ and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown and PICS_CMN_CMN_ChargingMode := dC
75	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_DC_WeldingDetection := unknown and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := true and PIXIT_CMN_CMN_Renegotiation := unknown and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PIXIT_EVCC_CMN_NewSAScheduleTupleID := newSAScheduleTupleID and PICS_CMN_CMN_ChargingMode := dC
76	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_Renegotiation := false and PIXIT_CMN_CMN_Renegotiation := renegotiation and PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown and PICS_CMN_EVCC_SalesTariff := true and PIXIT_EVCC_CMN_NewSAScheduleTupleID := newSAScheduleTupleID and PICS_CMN_CMN_ChargingMode := dC
77	PICS_CMN_CMN_IdentificationMode := eIM and	PIXIT_EVCC_DC_WeldingDetection :=

Configuration	EIM	PnC
	PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_DC_WeldingDetection := none_ and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := dC	weldingDetection and PICS_CMN_CMN_ChargingMode := dC
78	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_DC_WeldingDetection := unknown and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := dC	PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PICS_CMN_CMN_ChargingMode := dC
79	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PIXIT_EVCC_CMN_Pause := none_ and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := dC
80	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_DC_WeldingDetection := none_ and PIXIT_EVCC_CMN_Pause := none_ and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_DC_WeldingDetection := none_ and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := dC
81	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_DC_WeldingDetection := unknown and PIXIT_EVCC_CMN_Pause := none_ and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_DC_WeldingDetection := unknown and PIXIT_EVCC_CMN_Pause := unknown and PICS_CMN_CMN_ChargingMode := dC
82	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PIXIT_EVCC_CMN_Pause := none_ and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PIXIT_EVCC_CMN_Pause := none_ and PICS_CMN_CMN_ChargingMode := dC
83	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_DC_WeldingDetection := none_ and PIXIT_EVCC_CMN_Pause := none_ and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_DC_WeldingDetection := none_ and PIXIT_EVCC_CMN_Pause := none_ and PICS_CMN_CMN_ChargingMode := dC
84	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_DC_WeldingDetection := unknown and PIXIT_EVCC_CMN_Pause := none_ and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_DC_WeldingDetection := unknown and PIXIT_EVCC_CMN_Pause := none_ and PICS_CMN_CMN_ChargingMode := dC
85	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PIXIT_EVCC_CMN_Pause := none_ and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_ChargingMode := dC
86	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_DC_WeldingDetection := none_ and PIXIT_EVCC_CMN_Pause := none_ and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_DC_WeldingDetection := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_ChargingMode := dC
87	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_DC_WeldingDetection := unknown and PIXIT_EVCC_CMN_Pause := none_ and	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_DC_WeldingDetection := unknown and PIXIT_EVCC_CMN_Pause := pause and

Configuration	EIM	PnC
	PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_ChargingMode := dC
88	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := dC
89	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_DC_WeldingDetection := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_DC_WeldingDetection := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := dC
90	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_DC_WeldingDetection := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_DC_WeldingDetection := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := dC
91	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := dC
92	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_DC_WeldingDetection := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_DC_WeldingDetection := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := dC
93	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_DC_WeldingDetection := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_IdentificationMode := pnC and PIXIT_EVCC_DC_WeldingDetection := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := dC
94	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_PkiType := PKI-1 and PIXIT_CMN_CMN_PkiType := PKI-1 and PICS_CMN_CMN_ChargingMode := dC
95	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_DC_WeldingDetection := none_ and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_PkiType := PKI-2 and PIXIT_CMN_CMN_PkiType := PKI-2 and PICS_CMN_CMN_ChargingMode := dC
96	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_DC_WeldingDetection := unknown and PIXIT_EVCC_CMN_Pause := pause and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_PkiType := PKI-3 and PIXIT_CMN_CMN_PkiType := PKI-3 and PICS_CMN_CMN_ChargingMode := dC
97	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_PkiType := PKI-4 and PIXIT_CMN_CMN_PkiType := PKI-4 and PICS_CMN_CMN_ChargingMode := dC

Configuration	EIM	PnC
98	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tls and PIXIT_EVCC_DC_WeldingDetection := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_PkiType := PKI-5 and PIXIT_CMN_CMN_PkiType := PKI-5 and PICS_CMN_CMN_ChargingMode := dC
99	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tls and PIXIT_EVCC_DC_WeldingDetection := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_PkiType := PKI-6 and PIXIT_CMN_CMN_PkiType := PKI-6 and PICS_CMN_CMN_ChargingMode := dC
100	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_PkiType := PKI-7 and PIXIT_CMN_CMN_PkiType := PKI-7 and PICS_CMN_CMN_ChargingMode := dC
101	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_DC_WeldingDetection := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := dC	PICS_CMN_CMN_ChargingMode := dC
102	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_DC_WeldingDetection := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := dC	
103	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := dC	
104	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_DC_WeldingDetection := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := dC	
105	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_DC_WeldingDetection := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge and PICS_CMN_CMN_ChargingMode := dC	
106	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tls and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := dC	

Configuration	EIM	PnC
107	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_DC_WeldingDetection := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := dC	
108	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := tIs and PIXIT_EVCC_DC_WeldingDetection := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := dC	
109	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := dC	
110	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_DC_WeldingDetection := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := dC	
111	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := none_ and PIXIT_EVCC_DC_WeldingDetection := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := dC	
112	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_DC_WeldingDetection := weldingDetection and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := dC	
113	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_DC_WeldingDetection := none_ and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := dC	
114	PICS_CMN_CMN_IdentificationMode := eIM and PIXIT_EVCC_CMN_TLS := unknown and PIXIT_EVCC_DC_WeldingDetection := unknown and PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge and PICS_CMN_CMN_ChargingMode := dC	

7.4 Test suite identifiers

The selection of common naming conventions is one simple and often used mechanism to implement test suites which are consistent, maintainable and understandable for multiple users. Therefore, based on common ETSI naming conventions more specific naming conventions for the ISO 15118 conformance test suite are defined.

7.4.1 Module identifiers

All modules defined in the TSS shall start with a capital letter. The ISO 15118 test suite specific module identifier for template, function or test case modules is defined as:

<modtype>_<sut>_<ctx>

The segments of this identifier are defined in Table 16. An example for test case module identifier is:

TestCases_SECC_SupportedAppProtocol

Table 16 — ISO 15118 test suite naming convention for modules

Identifier	Values	Description
<modtype>		Module type
	TestCases	Module including test cases
	Functions	Module including functions
	Templates	Module including templates
<sut>		System under test
	EVCC	Electric Vehicle Communication Controller
	SECC	Supply Equipment Communication Controller
	CMN	Common (exclusively for template modules)
<ctx>	{fullname}	Context (e.g. name of message pattern signal name according to standard)

NOTE For module types other than templates, functions or test cases there is no identifier format defined.

7.4.2 Test case identifiers

The naming conventions for test cases are using a prefix, which is defined by ETSI as shown in Table 17.

Table 17 — ETSI naming convention for test case names

Keyword	Definition	Example
testcase name	Every testcase begins with TC (TC_) TC_TestCaseName	TC_SupportedAppProtocol

The ISO 15118 test suite specific test case identifier is defined as:

TC_<sut>_<dom>_<ttyp>_<ctx>_<nn>

The segments of this identifier are described in Table 18. An example for test case identifier is:

TC_SECC_AC_VTB_SupportedAppProtocol_001

Table 18 — ISO 15118 test suite naming convention for test case identifiers

Identifier	Values	Description
<prefix>	TC	see Table 17
<sut>		System under test
	EVCC	Electric Vehicle Communication Controller
	SECC	Supply Equipment Communication Controller
<dom>		Domain
	AC	AC specific behaviors

	DC	DC specific behaviors
	CMN	Common
<ttyp>		Type of testing
	VTB	Valid test behavior
	ITB	Invalid test behavior
<ctx>	{fullname}	Context (e.g. name of message pattern signal name according to standard)
<nn>	{xxx}	Sequential number from 001 to 999

7.4.3 Template identifiers

The naming conventions for templates are using a prefix, which is defined by ETSI as shown in Table 19.

Table 19 — ETSI naming convention for templates

Keyword	Context	Definition	Example
template name	Templates with concrete attribute values	Every template begins with keyword m (m_)	m_DNSRequest
template name	Templates with wildcards or matching expression	If a template contains or refers to templates with wildcards {*[?]} then template name begins with keyword mw (mw_)	mw_DNSResponse
template name	Templates with parameters, which do not assign or refer to templates with wildcards or matching expression	If a template contains attributes which are defined by parameters or constant values, then template name begins with keyword md (md_)	md_DNSResponse (integer ip)
template name	Templates with parameters, which do assign or refer to templates with wildcards or matching expression	If a template contains attributes which are defined by parameters, constant values or wildcards, then template name begins with keyword mdw (mdw_)	mdw_DNSResponse (integer ip)

The ISO 15118 test suite specific template identifier is defined as:

<prefix>_<sut>_<dom>_<dtyp>_<nn>

The segments of this identifier are described in Table 20. An example for template identifier is:

mdw_EVCC_CMN_SupportedAppProtocol_001

Table 20 — ISO 15118 test suite naming convention for template identifiers

Identifier	Values	Description
<prefix>		Type of template (see Table 19)
<sut>		System under test
	EVCC	Electric Vehicle Communication Controller
	SECC	Supply Equipment Communication Controller
	CMN	Common
<dom>		Domain

	AC	AC specific behaviors
	DC	DC specific behaviors
	CMN	Common
<dtyp>	{fullname}	Label of (root) data type according to standard
<nn>	{xxx}	Sequential number from 001 to 999

7.4.4 Function identifiers

The naming conventions for functions are using a prefix, which is defined by ETSI as shown in Table 21.

Table 21 — ETSI naming convention for function names

Keyword	Context	Definition	Example
function name	All functions	Every function begins with f (f_)	f_functionName

The ISO 15118 test suite naming convention for test case functions is defined as:

<prefix>_<sut>_<dom>_<ttyp>_<ctx>_<nn>

The segments of this identifier are described in Table 22. An example for template identifier is:

f_EVCC_AC_VTB_SessionSetup_001

Table 22 — ISO 15118 test suite naming convention for function names

Identifier	Values	Description
<prefix>		f_ (see Table 21)
<sut>		System under test
	EVCC	Electric Vehicle Communication Controller
	SECC	Supply Equipment Communication Controller
<dom>		Domain
	AC	AC specific behaviors
	DC	DC specific behaviors
	CMN	Common
<ttyp>		Type of testing
	VTB	Valid test behavior
	ITB	Invalid test behavior
<ctx>	{fullname}	Context (e.g. name of message pattern signal name according to standard)
<nn>	{xxx}	Sequential number from 001 to 999

7.4.5 Timer identifiers

The naming conventions for timers are using a prefix, which is defined by ETSI as shown in Table 23.

Table 23 — ETSI naming convention for timers

Keyword	Context	Definition	Example

timer name	t_	Local timer	t_wait
timer name	tc_	Timer defined within a component	tc_authMin

The ISO 15118 test suite specific timer identifier is defined as:

<prefix>_<ctx>

The segments of this identifier are described in Table 24. An example for timer identifier is:

tc_V2G_EVCC_Msg_Timer

Table 24 — ISO 15118 test suite naming convention for timer identifiers

Identifier	Values	Description
<prefix>		Type of timer (see Table 23)
<ctx>	{fullname}	Context (e.g. name of timer according to ISO 15118-2 or if not part of the standard any given name describing the context of the timer, e.g. EVCC_SDP_Timer)

7.4.6 PICS/PIXIT identifiers

The ISO 15118 test suite naming convention for PICS/PIXIT is defined as:

<pic>_<sut>_<dom>_<ctx>

The segments of this identifier are described in Table 25. An example for PICS/PIXIT identifier is:

PICS_CMN_CMN_IdentificationMode

Table 25 — ISO 15118 test suite naming convention for PICS/PIXIT identifiers

Identifier	Values	Description
<pic>		Protocol implementation capability
	PICS	Protocol Implementation Conformance Statement
	PIXIT	Protocol Implementation extra Information for Testing
<sut>		System under test
	EVCC	Electric Vehicle Communication Controller
	SECC	Supply Equipment Communication Controller
	CMN	Common
<dom>		Domain
	AC	AC specific behaviors
	DC	DC specific behaviors
	CMN	Common
<ctx>	{fullname}	Context (e.g. name of message pattern signal name according to standard)

7.4.7 Verdict identifiers

In this subclause the conventions for test verdicts are defined. The test verdicts defined in this document are listed in Table 26.

Table 26 — ISO 15118 test suite conventions on verdict handling

Verdict type	TTCN-3 Definition	ISO 15118 Test Suite
none	Is implicitly assigned in the beginning of every test case by default and is reported as a final verdict in the absence of any other verdict assignment during the test case execution.	No TSS specific definition (see TTCN-3 definition).
pass	Means that everything is OK. A verdict given when the observed outcome satisfies the test purpose and is valid with respect to the relevant requirements and with respect to the PICS. [ITU-T X.290]	If in review of a requirement the SUT has a correct behavior, then this verdict type shall be used.
inconc	A verdict given when the observed outcome is valid with respect to the relevant requirements but prevents the test purpose from being accomplished. [ITU-T X.290]	Means that neither pass nor fail can be reliably assigned.
fail	A verdict given when the observed outcome is syntactically invalid or inopportune with respect to the relevant requirements or the PICS/PIXIT. [ITU-T X.290]	If in review of a requirement the SUT has a wrong behavior, then this verdict type shall be used.

7.5 Test suite coverage

The following conditions apply in terms of test case coverage for this document regarding requirements defined in ISO 15118-2.

- Requirements that are out of scope according to the conventions defined in this document or cannot be tested in black-box tests are summarized in Table 27.
- Requirements that are not explicitly tested through a dedicated test case in the ATS but implicitly tested by the SUT Adapter/Codec are summarized in Table 28.

Table 27 — Requirements of ISO 15118-2 not considered in the ATS

List of requirement IDs	Comment (no consideration in ATS)
[V2G2-133], [V2G2-162], [V2G2-439], [V2G2-449], [V2G2-663], [V2G2-678], [V2G2-698], [V2G2-703], [V2G2-707], [V2G2-709], [V2G2-719], [V2G2-720], [V2G2-734], [V2G2-735], [V2G2-763], [V2G2-784], [V2G2-801], [V2G2-823], [V2G2-870], [V2G2-874], [V2G2-880], [V2G2-882], [V2G2-886], [V2G2-892], [V2G2-897], [V2G2-907], [V2G2-908], [V2G2-910], [V2G2-927], [V2G2-928]	Cannot be tested explicitly in EVCC black box test configuration
[V2G2-020], [V2G2-023], [V2G2-076], [V2G2-305], [V2G2-306], [V2G2-446], [V2G2-447], [V2G2-448]	Cannot be tested explicitly in EVCC black box test configuration due to strong configuration dependency on the SUT.
[V2G2-332], [V2G2-335], [V2G2-615], [V2G2-616]	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Out of scope of this document (PIXIT consideration leads to strong increase of TC quantity) for the

List of requirement IDs	Comment (no consideration in ATS)
	profile defined in the respective column.
[V2G2-134], [V2G2-146], [V2G2-219], [V2G2-233], [V2G2-234], [V2G2-301], [V2G2-317], [V2G2-322], [V2G2-325], [V2G2-328], [V2G2-336], [V2G2-337], [V2G2-338], [V2G2-444], [V2G2-540], [V2G2-542], [V2G2-544], [V2G2-546], [V2G2-547], [V2G2-549], [V2G2-550], [V2G2-552], [V2G2-553], [V2G2-555], [V2G2-556], [V2G2-558], [V2G2-559], [V2G2-561], [V2G2-562], [V2G2-564], [V2G2-566], [V2G2-567], [V2G2-569], [V2G2-570], [V2G2-572], [V2G2-574], [V2G2-578], [V2G2-579], [V2G2-581], [V2G2-583], [V2G2-585], [V2G2-586], [V2G2-588], [V2G2-589], [V2G2-592], [V2G2-595], [V2G2-596], [V2G2-598], [V2G2-629], [V2G2-639], [V2G2-668], [V2G2-690], [V2G2-726], [V2G2-727], [V2G2-774], [V2G2-796], [V2G2-798], [V2G2-803], [V2G2-808], [V2G2-824], [V2G2-826], [V2G2-853], [V2G2-857], [V2G2-872], [V2G2-877], [V2G2-883], [V2G2-887], [V2G2-902], [V2G2-916], [V2G2-923]	Cannot be tested explicitly in SECC black box test configuration
[V2G2-074], [V2G2-430], [V2G2-722], [V2G2-785], [V2G2-786], [V2G2-805]	Cannot be tested explicitly in SECC black box test configuration due to strong configuration dependency on the SUT.
[V2G2-011], [V2G2-012], [V2G2-089], [V2G2-090], [V2G2-092], [V2G2-094], [V2G2-096], [V2G2-135], [V2G2-640], [V2G2-760], [V2G2-830], [V2G2-835], [V2G2-868], [V2G2-869], [V2G2-878], [V2G2-911], [V2G2-925]	Cannot be tested explicitly in SECC/EVCC black box test configuration
[V2G2-867]	Cannot be tested explicitly in SECC/EVCC black box test configuration .
[V2G2-280], [V2G2-281]	Cannot be tested explicitly in SECC/EVCC black box test configuration due to strong configuration dependency on the SUT.
[V2G2-318], [V2G2-324], [V2G2-333], [V2G2-334], [V2G2-339], [V2G2-340], [V2G2-749], [V2G2-755], [V2G2-772], [V2G2-775], [V2G2-802], [V2G2-806], [V2G2-807]	Out of scope of this document (PIXIT consideration leads to strong increase of TC quantity)
[V2G2-001], [V2G2-002], [V2G2-003], [V2G2-014], [V2G2-016], [V2G2-017], [V2G2-037], [V2G2-038]	Out of scope of this document (Requirement refers to external RFCs, standards, etc.)

List of requirement IDs	Comment (no consideration in ATS)
[V2G2-039], [V2G2-040], [V2G2-041], [V2G2-042], [V2G2-043], [V2G2-044], [V2G2-045], [V2G2-046], [V2G2-047], [V2G2-049], [V2G2-050], [V2G2-051], [V2G2-052], [V2G2-053], [V2G2-054], [V2G2-055], [V2G2-057], [V2G2-058], [V2G2-059], [V2G2-060], [V2G2-061], [V2G2-062], [V2G2-063], [V2G2-064], [V2G2-065], [V2G2-067], [V2G2-070], [V2G2-071], [V2G2-163], [V2G2-164], [V2G2-638], [V2G2-649], [V2G2-650], [V2G2-693], [V2G2-694], [V2G2-695], [V2G2-810], [V2G2-811], [V2G2-843], [V2G2-844], [V2G2-850], [V2G2-873], [V2G2-876], [V2G2-885], [V2G2-924]	
[V2G2-289], [V2G2-290], [V2G2-291], [V2G2-292], [V2G2-314], [V2G2-315], [V2G2-329], [V2G2-330], [V2G2-480], [V2G2-643], [V2G2-644], [V2G2-674], [V2G2-676], [V2G2-743], [V2G2-761], [V2G2-777], [V2G2-833], [V2G2-834], [V2G2-837], [V2G2-838], [V2G2-851], [V2G2-858], [V2G2-859], [V2G2-860], [V2G2-862], [V2G2-863], [V2G2-865], [V2G2-866]	Out of scope of this document (Requirement refers to power flow)
[V2G2-035], [V2G2-036]	Out of scope of this document (Requirements refers to ISO 15118-5)

Table 28 — Requirements of ISO 15118-2 implicitly covered by the SUT adapters / codecs

List of requirement IDs	Comment (implicitly tested)
[V2G2-010]	Cannot be tested explicitly in SECC/EVCC black box test configuration for the profile defined in the respective column. Tested implicitly by V2G codec for the profile defined in the respective column
[V2G2-073], [V2G2-075], [V2G2-077], [V2G2-078], [V2G2-079], [V2G2-080], [V2G2-081], [V2G2-083], [V2G2-084], [V2G2-085], [V2G2-124], [V2G2-125], [V2G2-128], [V2G2-129], [V2G2-131], [V2G2-132], [V2G2-136], [V2G2-137], [V2G2-138], [V2G2-139], [V2G2-143], [V2G2-149], [V2G2-151], [V2G2-155], [V2G2-156], [V2G2-174], [V2G2-622], [V2G2-809], [V2G2-816]	Tested implicitly by 15118-2 SUT adapter
[V2G2-097], [V2G2-098], [V2G2-099], [V2G2-100], [V2G2-101], [V2G2-102],	Tested implicitly by V2G codec

List of requirement IDs	Comment (implicitly tested)
[V2G2-171], [V2G2-440], [V2G2-445], [V2G2-600], [V2G2-697], [V2G2-825]	
[V2G2-651]	Tested implicitly by V2G codec. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column.

The resulting test suite coverage in this document with reference to requirements in ISO 15118-2 is summarized in Table 29. It only includes requirements which are not listed for exclusion in Table 27 or Table 28. Table 29 defines the relevance of requirements of ISO 15118-2 for the ATS, based upon the type of SUT (EVCC or SECC), the authentication profile (EIM or PnC) and AC or DC charging. The following symbols are used in Table 29.

- X Indicates requirements that are covered in the ATS with one or more test cases.
- I Indicates requirements that are indirectly covered in the ATS but other than those listed in Table 28 can be directly associated to one or more test cases in the ATS.
- P Indicates requirements that are only partially covered by one or more test cases in the ATS.
- N Requirements that are not testable for the profile defined in the respective column.
- Requirements that are not applicable for the profile defined in the respective column.
- O Requirements that are out of scope for the profile defined in the respective column.

Table 29 — ATS coverage of requirements in ISO 15118-2

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-004]	N	N	X	X	N	N	X	X	Cannot be tested explicitly in SECC/EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]
[V2G2-005]	—	—	P	P	—	—	X	X	The verification of this requirement consists of two different test cases, so the requirement can only be tested partially for the profile defined in the respective column Group_[V2G2-005]_[V2G2-007]
[V2G2-006]	—	—	—	—	—	—	X	X	Group_[V2G2-006]_[V2G2-652]_[V2G2-909]
[V2G2-007]	—	—	P	P	—	—	X	X	The verification of this requirement consists of two different test cases, so the requirement can only be tested partially for the profile defined in the respective column Group_[V2G2-005]_[V2G2-007]
[V2G2-008]	—	—	—	—	I	I	I	I	Tested indirectly through test for requirement [V2G2-166]. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column TC_EVCC_CMN_VTB_SupportedAppProtocol_001
[V2G2-009]	N	N	X	X	N	N	X	X	Cannot be tested explicitly in SECC/EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]
[V2G2-018]	—	—	—	—	I	I	I	I	Tested indirectly through test for requirement [V2G2-142]

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									Group_[V2G2-142]_[V2G2-623]_[V2G2-018]_[V2G2-019]
[V2G2-019]	—	—	—	—	I	I	I	I	Tested indirectly through test for requirement [V2G2-142] Group_[V2G2-142]_[V2G2-623]_[V2G2-018]_[V2G2-019]
[V2G2-021]	—	—	—	—	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column TC_EVCC_CMN_VTB_SupportedAppProtocol_001
[V2G2-022]	—	—	—	—	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-024]	—	—	—	—	X	X	X	X	TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-025]	—	—	—	—	X	X	X	X	Group_[V2G2-728]_[V2G2-025]
[V2G2-026]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-168] TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_004 TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-027]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-027] TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_003 TC_SECC_CMN_VTB_SDP_007
[V2G2-029]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-723] TC_SECC_CMN_VTB_SDP_004 TC_SECC_CMN_VTB_SDP_005 TC_SECC_CMN_VTB_SDP_006
[V2G2-030]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-168] TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_004 TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-031]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-168] TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_004 TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-032]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-032] TC_SECC_CMN_VTB_SupportedAppProtocol_005
[V2G2-033]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-168] TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_004

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-034]	X	X	X	X	—	—	—	—	Group_[V2G2-729]_[V2G2-034]
[V2G2-068]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-168]. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_004 TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-082]	X	X	X	X	X	X	X	X	Group_[V2G2-082]_[V2G2-086]
[V2G2-086]	X	X	X	X	X	X	X	X	Group_[V2G2-082]_[V2G2-086]
[V2G2-087]	P	P	P	P	P	P	P	P	Table 9 and Table 10 can only be tested partially TC_EVCC_CMN_VTB_V2GTPSDP_001 TC_EVCC_CMN_VTB_V2GTPSessionSetup_001 TC_SECC_CMN_VTB_V2GTPSDP_001 TC_SECC_CMN_VTB_V2GTPSessionSetup_001
[V2G2-088]	P	P	P	P	P	P	P	P	Requirement definition considers various possible scenarios, so the requirement can only be tested partially for exemplarily selected cases TC_EVCC_CMN_VTB_V2GTPSessionSetup_001 TC_SECC_CMN_VTB_V2GTPSessionSetup_001
[V2G2-103]	—	—	X	X	—	—	X	X	Group_[V2G2-103]_[V2G2-104]
[V2G2-104]	—	—	P	P	—	—	P	P	Internal states cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-103]_[V2G2-104]
[V2G2-108]	—	—	—	—	—	—	X	X	TC_EVCC_CMN_VTB_PaymentDetails_001 TC_EVCC_CMN_VTB_PaymentDetails_002
[V2G2-117]	—	—	—	—	—	—	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-119]	—	—	—	—	—	—	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-121]	—	—	—	—	—	—	I	I	Tested indirectly through test for requirement [V2G2-500]/[V2G2-501] and [V2G2-503] TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011
[V2G2-122]	—	—	—	—	—	—	I	I	Tested indirectly through test for requirement [V2G2-500]/[V2G2-501] and [V2G2-503] TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011
[V2G2-123]	X	X	X	X	—	—	—	—	TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_003 TC_SECC_CMN_VTB_SDP_007
[V2G2-126]	—	—	—	—	X	X	X	X	TC_EVCC_CMN_VTB_SDP_006

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-127]	—	—	—	—	P	P	P	P	Subclause 7.8.3.1 can only be tested partially TC_EVCC_CMN_VTB_V2GTPSDP_001
[V2G2-130]	P	P	P	P	—	—	—	—	Subclause 7.8.3.1 can only be tested partially TC_SECC_CMN_VTB_V2GTPSDP_001
[V2G2-140]	—	—	—	—	X	X	X	X	TC_EVCC_CMN_VTB_V2GTPSDP_001
[V2G2-141]	—	—	—	—	X	X	X	X	TC_EVCC_CMN_VTB_V2GTPSDP_001
[V2G2-142]	—	—	—	—	X	X	X	X	Group_[V2G2-142]_[V2G2-623]_[V2G2-018]_[V2G2-019]
[V2G2-144]	P	P	P	P	—	—	—	—	Requirement definition considers various possible scenarios, so the requirement can only be tested partially for exemplarily selected cases TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_003 TC_SECC_CMN_VTB_SDP_007
[V2G2-145]	—	—	—	—	P	P	P	P	Requirement definition considers various possible scenarios, so the requirement can only be tested partially for exemplarily selected cases TC_EVCC_CMN_VTB_SDP_007
[V2G2-147]	X	X	X	X	—	—	—	—	TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_003 TC_SECC_CMN_VTB_SDP_007
[V2G2-150]	X	X	X	X	—	—	—	—	TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_003 TC_SECC_CMN_VTB_SDP_007
[V2G2-152]	X	X	X	X	—	—	—	—	TC_SECC_CMN_VTB_V2GTPSDP_001
[V2G2-153]	X	X	X	X	—	—	—	—	TC_SECC_CMN_VTB_V2GTPSDP_001
[V2G2-154]	X	X	X	X	—	—	—	—	TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_003 TC_SECC_CMN_VTB_SDP_007
[V2G2-157]	—	—	—	—	I	I	I	I	Tested indirectly through test for requirement [V2G2-161] Group_[V2G2-157]_[V2G2-160]_[V2G2-161]
[V2G2-158]	—	—	—	—	I	I	I	I	Tested indirectly through test for requirement [V2G2-157] and an initiated paused session. Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-158]_[V2G2-731]
[V2G2-159]	—	—	—	—	I	I	I	I	Tested indirectly through test for requirement [V2G2-160] Group_[V2G2-159]
[V2G2-160]	—	—	—	—	X	X	X	X	Group_[V2G2-157]_[V2G2-160]_[V2G2-161]
[V2G2-161]	—	—	—	—	I	I	I	I	Group_[V2G2-157]_[V2G2-160]_[V2G2-161] Group_[V2G2-161]
[V2G2-165]	X	X	X	X	X	X	X	X	Group_[V2G2-165]_[V2G2-178]

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-166]	—	—	—	—	X	X	X	X	TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-167]	—	—	—	—	X	X	X	X	TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-168]	X	X	X	X	—	—	—	—	TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_004 TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-169]	P	P	P	P	—	—	—	—	Priority handling cannot be tested explicitly in SECC black box test configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_004 TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-170]	X	X	X	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-172]	X	X	X	X	—	—	—	—	TC_SECC_CMN_VTB_SupportedAppProtocol_002
[V2G2-173]	—	—	—	—	X	X	X	X	TC_EVCC_CMN_VTB_SupportedAppProtocol_002
[V2G2-175]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-176]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_004 TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-178]	P	P	P	P	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-165]_[V2G2-178]
[V2G2-179]	X	X	X	X	X	X	X	X	Group_[V2G2-179]_[V2G2-180]_[V2G2-181]_[V2G2-183]
[V2G2-180]	X	X	X	X	X	X	X	X	Group_[V2G2-179]_[V2G2-180]_[V2G2-181]_[V2G2-183]
[V2G2-181]	X	X	X	X	X	X	X	X	Group_[V2G2-179]_[V2G2-180]_[V2G2-181]_[V2G2-183]
[V2G2-182]	X	X	X	X	X	X	X	X	Group_[V2G2-182]
[V2G2-183]	X	X	X	X	X	X	X	X	Group_[V2G2-179]_[V2G2-180]_[V2G2-181]_[V2G2-183]
[V2G2-188]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-188]_[V2G2-189]_[V2G2-485]
[V2G2-189]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-188]_[V2G2-189]_[V2G2-485]

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-190]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-190]_[V2G2-191]_[V2G2-543]_[V2G2-192]
[V2G2-191]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-190]_[V2G2-191]_[V2G2-543]_[V2G2-192]
[V2G2-192]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-190]_[V2G2-191]_[V2G2-543]_[V2G2-192]
[V2G2-193]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_ServiceDiscovery_001
[V2G2-194]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_ServiceDiscovery_001
[V2G2-195]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-195]_[V2G2-196]
[V2G2-196]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-195]_[V2G2-196]
[V2G2-197]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_ServiceDetail_001 TC_EVCC_CMN_VTB_ServiceDetail_002
[V2G2-198]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_ServiceDetail_001 TC_EVCC_CMN_VTB_ServiceDetail_002
[V2G2-199]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-199]_[V2G2-200]
[V2G2-200]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-199]_[V2G2-200]
[V2G2-201]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]
[V2G2-202]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]
[V2G2-203]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-203]_[V2G2-204]

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-204]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-203]_[V2G2-204]
[V2G2-205]	—	—	N	N	—	—	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001 TC_EVCC_CMN_VTB_PaymentDetails_001 TC_EVCC_CMN_VTB_PaymentDetails_002 TC_EVCC_CMN_VTB_PaymentDetails_003
[V2G2-206]	—	—	N	N	—	—	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001 TC_EVCC_CMN_VTB_PaymentDetails_001 TC_EVCC_CMN_VTB_PaymentDetails_002 TC_EVCC_CMN_VTB_PaymentDetails_003
[V2G2-208]	—	—	X	X	—	—	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-208]_[V2G2-209]
[V2G2-209]	—	—	X	X	—	—	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-208]_[V2G2-209]
[V2G2-210]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_Authorization_001 TC_EVCC_CMN_VTB_Authorization_002 TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011
[V2G2-211]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_Authorization_001 TC_EVCC_CMN_VTB_Authorization_002 TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011
[V2G2-212]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-212]_[V2G2-213]
[V2G2-213]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-212]_[V2G2-213]
[V2G2-214]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-214]_[V2G2-216]
[V2G2-215]	—	—	—	—	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered. Group_[V2G2-215]
[V2G2-216]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									Group_[V2G2-214]_[V2G2-216]
[V2G2-218]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-218]_[V2G2-220]
[V2G2-220]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-218]_[V2G2-220]
[V2G2-221]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-221]_[V2G2-222]_[V2G2-286]
[V2G2-222]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-221]_[V2G2-222]_[V2G2-286]
[V2G2-223]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-223]_[V2G2-226]
[V2G2-224]	X	X	X	X	—	—	—	—	Requirement definition considers various possible scenarios, so the requirement can only be tested partially for exemplarily selected cases TC_SECC_AC_VTB_PowerDelivery_001 TC_SECC_AC_VTB_PowerDelivery_002 TC_SECC_DC_VTB_PowerDelivery_001
[V2G2-225]	X	X	X	X	—	—	—	—	TC_SECC_AC_VTB_PowerDelivery_010 TC_SECC_DC_VTB_PowerDelivery_007
[V2G2-226]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-223]_[V2G2-226]
[V2G2-227]	—	—	—	—	—	—	X	X	TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-228]	—	—	N	N	—	—	X	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-229]	—	—	N	N	—	—	X	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-230]	—	—	—	—	—	—	I	I	Tested indirectly through test for requirement [V2G2-501] and [V2G2-503] TC_EVCC_CMN_VTB_Authorization_011
[V2G2-231]	—	—	X	X	—	—	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-231]_[V2G2-232]
[V2G2-232]	—	—	X	X	—	—	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-231]_[V2G2-232]
[V2G2-235]	—	—	N	N	—	—	X	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									for the profile defined in the respective column. TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-236]	—	—	N	N	—	—	X	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-237]	—	—	X	X	—	—	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-237]_[V2G2-238]
[V2G2-238]	—	—	X	X	—	—	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-237]_[V2G2-238]
[V2G2-239]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-239]_[V2G2-738]
[V2G2-240]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-240]_[V2G2-241]
[V2G2-241]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-240]_[V2G2-241]
[V2G2-242]	N	—	N	—	X	—	X	—	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-242]
[V2G2-243]	X	—	X	—	N	—	N	—	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-243]_[V2G2-244]
[V2G2-244]	X	—	X	—	N	—	N	—	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-243]_[V2G2-244]
[V2G2-245]	—	—	N	N	—	—	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-245]_[V2G2-246]
[V2G2-246]	—	—	N	N	—	—	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-245]_[V2G2-246]
[V2G2-247]	—	—	X	X	—	—	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-247]_[V2G2-248]
[V2G2-248]	—	—	X	X	—	—	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-247]_[V2G2-248]
[V2G2-249]	—	N	—	N	—	X	—	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_DC_VTB_CableCheck_001

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-250]	—	N	—	N	—	X	—	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_DC_VTB_CableCheck_001
[V2G2-251]	—	X	—	X	—	N	—	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-251]_[V2G2-252]
[V2G2-252]	—	X	—	X	—	N	—	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-251]_[V2G2-252]
[V2G2-253]	—	N	—	N	—	X	—	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-253]_[V2G2-254]
[V2G2-254]	—	N	—	N	—	X	—	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-253]_[V2G2-254]
[V2G2-255]	—	X	—	X	—	N	—	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_DC_VTB_PreCharge_001 TC_SECC_DC_VTB_PreCharge_003
[V2G2-256]	—	X	—	X	—	N	—	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_DC_VTB_PreCharge_001 TC_SECC_DC_VTB_PreCharge_003
[V2G2-257]	—	N	—	N	—	X	—	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-257]_[V2G2-258]
[V2G2-258]	—	N	—	N	—	X	—	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-257]_[V2G2-258]
[V2G2-259]	—	X	—	X	—	N	—	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-259]_[V2G2-260]
[V2G2-260]	—	X	—	X	—	N	—	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-259]_[V2G2-260]
[V2G2-261]	—	N	—	N	—	X	—	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_DC_VTB_WeldingDetection_001 TC_EVCC_DC_VTB_WeldingDetection_009
[V2G2-262]	—	N	—	N	—	X	—	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_DC_VTB_WeldingDetection_001 TC_EVCC_DC_VTB_WeldingDetection_009
[V2G2-263]	—	X	—	X	—	N	—	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_DC_VTB_WeldingDetection_001

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_SECC_DC_VTB_WeldingDetection_003 TC_SECC_DC_VTB_WeldingDetection_005
[V2G2-264]	—	X	—	X	—	N	—	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_DC_VTB_WeldingDetection_001 TC_SECC_DC_VTB_WeldingDetection_003 TC_SECC_DC_VTB_WeldingDetection_005
[V2G2-265]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-266]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-267]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-268]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-271]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-272]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-274]	—	—	X	X	—	—	X	X	TC_EVCC_CMN_VTB_CertificateUpdate_001 TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-275]	—	—	X	X	—	—	X	X	TC_EVCC_CMN_VTB_CertificateUpdate_001 TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-276]	X	X	X	X	—	—	X	X	Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-276]_[V2G2-277]
[V2G2-277]	X	X	X	X	—	—	X	X	Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-276]_[V2G2-277]
[V2G2-278]	X	X	X	X	X	X	X	X	Group_[V2G2-278]_[V2G2-279]
[V2G2-279]	X	X	X	X	X	X	X	X	Group_[V2G2-278]_[V2G2-279]
[V2G2-282]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-283]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-284]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-836]_[V2G2-284]_[V2G2-606]_[V2G2-288]_[V2G2-607]
[V2G2-286]	—	—	—	—	P	X	P	X	Group_[V2G2-221]_[V2G2-222]_[V2G2-286]
[V2G2-288]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-836]_[V2G2-284]_[V2G2-606]_[V2G2-288]_[V2G2-607]
[V2G2-293]	—	—	—	—	X	X	X	X	Group_[V2G2-293]_[V2G2-673]
[V2G2-294]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-294]_[V2G2-608]_[V2G2-299]_[V2G2-609]_[V2G2-300]
[V2G2-296]	—	—	—	—	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-296]

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-297]	—	—	—	—	I	I	I	I	Tested indirectly through test for requirement [V2G2-298]. Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-297]_[V2G2-298]
[V2G2-298]	—	—	—	—	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-297]_[V2G2-298]
[V2G2-299]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-294]_[V2G2-608]_[V2G2-299]_[V2G2-609]_[V2G2-300]
[V2G2-300]	X	X	X	X	—	—	—	—	Group_[V2G2-294]_[V2G2-608]_[V2G2-299]_[V2G2-609]_[V2G2-300]
[V2G2-303]	P	P	P	P	—	—	—	—	SalesTariff is only testable if a corresponding PIXIT is considered Group_[V2G2-303]
[V2G2-304]	P	P	P	P	—	—	—	—	SalesTariff is only testable if a corresponding PIXIT is considered Group_[V2G2-304]
[V2G2-307]	—	—	X	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_[V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-308]	—	—	I	I	—	—	—	—	Tested indirectly through test of signature validation. Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_[V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-309]	—	—	I	I	—	—	—	—	Tested indirectly through test of signature validation. Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_[V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-310]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-313]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-316]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_[V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-321]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_[V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									321]_[V2G2-613]
[V2G2-327]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-331]	P	P	P	P	—	—	—	—	SalesTariff is only testable if a corresponding PIXIT is considered Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-343]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-344]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-345]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-346]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-347]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-348]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_002

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-349]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]
[V2G2-350]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]
[V2G2-351]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]
[V2G2-352]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]
[V2G2-353]	—	—	X	X	—	—	X	X	Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]
[V2G2-354]	—	—	X	X	—	—	X	X	Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]
[V2G2-355]	—	—	N	N	—	—	X	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_CertificateInstallation_001 TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-356]	—	—	N	N	—	—	X	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_CertificateInstallation_001 TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-358]	X	—	X	—	N	—	N	—	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-358]_[V2G2-359]
[V2G2-359]	X	—	X	—	N	—	N	—	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-358]_[V2G2-359]
[V2G2-360]	N	—	N	—	X	—	X	—	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-360]_[V2G2-361]
[V2G2-361]	N	—	N	—	X	—	X	—	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-360]_[V2G2-361]
[V2G2-362]	X	—	X	—	N	—	N	—	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-573]_[V2G2-362]_[V2G2-363]
[V2G2-363]	X	—	X	—	N	—	N	—	Cannot be tested explicitly in EVCC black box test configuration for

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									the profile defined in the respective column Group_[V2G2-573]_[V2G2-362]_[V2G2-363]
[V2G2-364]	—	X	—	X	—	N	—	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-364]_[V2G2-365]_[V2G2-366]
[V2G2-365]	—	X	—	X	—	N	—	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-364]_[V2G2-365]_[V2G2-366]
[V2G2-366]	—	P	—	P	—	-	—	—	Table 98 can only be tested partially Group_[V2G2-364]_[V2G2-365]_[V2G2-366]
[V2G2-367]	—	N	—	N	—	X	—	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-367]_[V2G2-368]_[V2G2-369]
[V2G2-368]	—	N	—	N	—	X	—	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-367]_[V2G2-368]_[V2G2-369]
[V2G2-369]	—	—	—	—	—	P	—	P	Table 100 can only be tested partially Group_[V2G2-367]_[V2G2-368]_[V2G2-369]
[V2G2-370]	—	N	—	N	—	X	—	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-370]_[V2G2-371]
[V2G2-371]	—	N	—	N	—	X	—	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-370]_[V2G2-371]
[V2G2-372]	—	X	—	X	—	N	—	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-372]_[V2G2-373]
[V2G2-373]	—	X	—	X	—	N	—	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-372]_[V2G2-373]
[V2G2-374]	—	N	—	N	—	X	—	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-374]_[V2G2-375]
[V2G2-375]	—	N	—	N	—	X	—	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-374]_[V2G2-375]
[V2G2-376]	—	—	—	—	P	—	—	—	Table 104 can only be tested partially Group_[V2G2-376]
[V2G2-377]	P	—	—	—	—	—	—	—	Table 104 can only be tested partially Group_[V2G2-377]
[V2G2-378]	—	—	—	—	X	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_ServiceDetail_002
[V2G2-379]	X	—	—	—	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-380]	—	—	—	—	—	—	P	—	Table 104 can only be tested partially Group_[V2G2-380]
[V2G2-381]	—	—	P	—	—	—	—	—	Table 104 can only be tested partially Group_[V2G2-381]
[V2G2-384]	—	—	—	—	—	—	X	—	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_ServiceDetail_001
[V2G2-385]	—	—	X	—	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-386]	—	—	—	—	—	—	X	—	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-387]	—	—	X	—	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_CertificateInstallation_001
[V2G2-388]	—	—	—	—	—	—	X	—	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-389]	—	—	X	—	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-390]	—	—	—	—	—	P	—	—	Table 104 can only be tested partially Group_[V2G2-390]
[V2G2-391]	—	P	—	—	—	—	—	—	Table 104 can only be tested partially Group_[V2G2-391]
[V2G2-392]	—	—	—	—	—	X	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_ServiceDetail_002
[V2G2-393]	—	X	—	—	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011
[V2G2-394]	—	—	—	—	—	—	—	P	Table 104 can only be tested partially Group_[V2G2-394]
[V2G2-395]	—	—	—	P	—	—	—	—	Table 104 can only be tested partially Group_[V2G2-395]
[V2G2-396]	—	—	—	—	—	—	—	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_ServiceDetail_001
[V2G2-397]	—	—	—	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-398]	—	—	—	—	—	—	—	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-399]	—	—	—	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_CertificateInstallation_001
[V2G2-400]	—	—	—	—	—	—	—	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-401]	—	—	—	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-402]	—	—	—	—	P	—	—	—	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-402]
[V2G2-403]	—	—	—	—	—	P	—	—	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-403]
[V2G2-404]	—	—	—	—	—	—	P	—	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-404]
[V2G2-405]	—	—	—	—	—	—	—	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-405]
[V2G2-406]	—	—	X	—	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_AC_VTB_ChargingStatus_002 TC_SECC_AC_VTB_ChargingStatus_007 TC_SECC_AC_VTB_ChargingStatus_009
[V2G2-407]	—	—	X	—	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_AC_VTB_ChargingStatus_001 TC_SECC_AC_VTB_ChargingStatus_006 TC_SECC_AC_VTB_ChargingStatus_008
[V2G2-408]	—	—	—	—	—	—	X	—	Group_[V2G2-512]_[V2G2-408]
[V2G2-410]	—	—	X	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_CertificateInstallation_001
[V2G2-411]	—	—	—	—	—	—	X	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-412]	—	—	X	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-413]	—	—	—	—	—	—	X	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-414]	X	X	X	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-415]	—	—	—	—	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_ServiceDetail_001 TC_EVCC_CMN_VTB_ServiceDetail_002
[V2G2-416]	P	P	P	P	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-417]	N	N	N	N	P	P	P	P	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_ServiceDetail_001 TC_EVCC_CMN_VTB_ServiceDetail_002
[V2G2-418]	X	X	X	X	—	—	—	—	TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-419]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirements [V2G2-420] and [V2G2-421] TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-420]	P	P	P	P	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-421]	P	P	P	P	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-422]	—	—	—	—	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_ServiceDetail_001 TC_EVCC_CMN_VTB_ServiceDetail_002
[V2G2-424]	—	—	—	—	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_ServiceDetail_001 TC_EVCC_CMN_VTB_ServiceDetail_002
[V2G2-425]	X	X	X	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_007 TC_SECC_CMN_VTB_ServiceDetail_008
[V2G2-426]	X	X	X	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-427]	X	X	X	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-428]	—	—	P	P	—	—	—	—	Table 106 can only be tested partially TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-429]	P	P	P	P	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered. Table 107 can only be tested partially TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-431]	—	—	—	—	X	X	X	X	Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]
[V2G2-432]	—	—	—	—	X	X	X	X	Group_[V2G2-493]_[V2G2-494]_[V2G2-432]
[V2G2-433]	X	X	X	X	—	—	—	—	TC_SECC_CMN_VTB_PaymentServiceSelection_006
[V2G2-434]	—	—	—	—	P	P	P	P	Table 108 and 109 can only be tested partially Group_[V2G2-434]_[V2G2-436]_[V2G2-437]_[V2G2-438]
[V2G2-435]	P	P	P	P	—	—	—	—	Table 108 and 109 can only be tested partially Group_[V2G2-435]
[V2G2-436]	—	—	—	—	I	I	I	I	Tested indirectly through test for requirement [V2G2-438] Group_[V2G2-434]_[V2G2-436]_[V2G2-437]_[V2G2-438]
[V2G2-437]	—	—	—	—	I	I	I	I	Tested indirectly through test for requirement [V2G2-438] Group_[V2G2-434]_[V2G2-436]_[V2G2-437]_[V2G2-438]
[V2G2-438]	—	—	—	—	P	P	P	P	Internal states cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-434]_[V2G2-436]_[V2G2-437]_[V2G2-438]
[V2G2-441]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-443] Group_[V2G2-441]_[V2G2-442]_[V2G2-443]
[V2G2-442]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-443] Group_[V2G2-441]_[V2G2-442]_[V2G2-443]
[V2G2-443]	P	P	P	P	—	—	—	—	Internal states cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-441]_[V2G2-442]_[V2G2-443]
[V2G2-457]	P	P	P	P	—	—	—	—	Only one message sequence will be considered, so the requirement can only be tested partially Group_[V2G2-457]
[V2G2-458]	X	X	X	X	—	—	—	—	Group_[V2G2-458]
[V2G2-459]	X	X	X	X	—	—	—	—	Only one message sequence will be considered, so the requirement can only be tested partially Group_[V2G2-459]_[V2G2-538]
[V2G2-460]	X	X	X	X	—	—	—	—	Only one message sequence will be considered, so the requirement can only be tested partially Group_[V2G2-460]
[V2G2-461]	—	—	X	X	—	—	—	—	Group_[V2G2-461]
[V2G2-462]	X	X	X	X	—	—	—	—	Group_[V2G2-462]
[V2G2-463]	X	X	X	X	—	—	—	—	TC_SECC_AC_VTB_SessionSetup_001 TC_SECC_AC_VTB_SessionSetup_004 TC_SECC_DC_VTB_SessionSetup_001

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_SECC_DC_VTB_SessionSetup_004
[V2G2-464]	X	X	X	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_007 TC_SECC_CMN_VTB_ServiceDetail_008
[V2G2-465]	X	X	X	X	—	—	—	—	TC_SECC_CMN_VTB_PaymentServiceSelection_009 TC_SECC_CMN_VTB_PaymentServiceSelection_010
[V2G2-467]	X	X	X	X	—	—	—	—	TC_SECC_CMN_VTB_PaymentServiceSelection_006
[V2G2-468]	—	—	X	X	—	—	—	—	TC_SECC_CMN_VTB_CertificateInstallation_007
[V2G2-470]	—	—	X	X	—	—	—	—	TC_SECC_CMN_VTB_CertificateUpdate_007 TC_SECC_CMN_VTB_CertificateUpdate_008 TC_SECC_CMN_VTB_CertificateUpdate_009 TC_SECC_CMN_VTB_CertificateUpdate_010 TC_SECC_CMN_VTB_CertificateUpdate_011
[V2G2-473]	—	—	X	X	—	—	—	—	TC_SECC_CMN_VTB_CertificateUpdate_016
[V2G2-474]	—	—	X	X	—	—	—	—	TC_SECC_CMN_VTB_PaymentDetails_007
[V2G2-475]	—	—	X	X	—	—	—	—	TC_SECC_CMN_VTB_Authorization_007
[V2G2-476]	X	X	X	X	—	—	—	—	TC_SECC_AC_VTB_ChargeParameterDiscovery_004 TC_SECC_DC_VTB_ChargeParameterDiscovery_003
[V2G2-477]	X	X	X	X	—	—	—	—	TC_SECC_AC_VTB_ChargeParameterDiscovery_005 TC_SECC_DC_VTB_ChargeParameterDiscovery_004
[V2G2-478]	X	X	X	X	—	—	—	—	TC_SECC_AC_VTB_PowerDelivery_010 TC_SECC_DC_VTB_PowerDelivery_007
[V2G2-479]	X	X	X	X	—	—	—	—	TC_SECC_AC_VTB_PowerDelivery_011 TC_SECC_DC_VTB_PowerDelivery_008
[V2G2-481]	—	—	X	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-481]
[V2G2-482]	—	—	—	—	P	P	P	P	Requirement definition considers various possible scenarios, so the requirement can only be tested partially for exemplarily selected cases Group_[V2G2-482]
[V2G2-483]	—	—	—	—	X	X	X	X	TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-484]	—	—	—	—	X	X	X	X	TC_EVCC_CMN_VTB_SupportedAppProtocol_003
[V2G2-485]	—	—	—	—	P	P	P	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-188]_[V2G2-189]_[V2G2-485]
[V2G2-486]	—	—	—	—	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially TC_EVCC_CMN_VTB_SessionSetup_002 TC_EVCC_CMN_VTB_SessionSetup_003 TC_EVCC_CMN_VTB_SessionSetup_004
[V2G2-487]	—	—	—	—	P	P	P	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_EVCC_CMN_VTB_ServiceDiscovery_001
[V2G2-488]	—	—	—	—	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially TC_EVCC_CMN_VTB_ServiceDiscovery_002 TC_EVCC_CMN_VTB_ServiceDiscovery_003 TC_EVCC_CMN_VTB_ServiceDiscovery_004 TC_EVCC_CMN_VTB_ServiceDiscovery_005
[V2G2-489]	—	—	—	—	P	P	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_ServiceDetail_001 TC_EVCC_CMN_VTB_ServiceDetail_002
[V2G2-490]	—	—	—	—	P	P	P	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-490]
[V2G2-491]	—	—	—	—	P	P	P	P	Requirement is only testable if a corresponding PIXIT is considered. Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-491]
[V2G2-492]	—	—	—	—	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-492]
[V2G2-493]	—	—	—	—	P	P	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-493]_[V2G2-494]_[V2G2-432]
[V2G2-494]	—	—	—	—	P	P	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-493]_[V2G2-494]_[V2G2-432]
[V2G2-495]	—	—	—	—	—	—	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_PaymentDetails_003
[V2G2-496]	—	—	—	—	—	—	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-497]	—	—	—	—	—	—	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-498]	—	—	—	—	—	—	P	P	Requirement is only testable if a corresponding PIXIT is considered. Only one case can be considered, so the requirement

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									can only be tested partially Group_[V2G2-498]
[V2G2-499]	—	—	—	—	—	—	P	P	Requirement is only testable if a corresponding PIXIT is considered. Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-499]
[V2G2-500]	—	—	—	—	—	—	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_PaymentDetails_001
[V2G2-501]	—	—	—	—	—	—	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_PaymentDetails_002
[V2G2-502]	—	—	—	—	—	—	P	P	Requirement is only testable if a corresponding PIXIT is considered. Only one case can be considered, so the requirement can only be tested partially TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_002 TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_003 TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_004 TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005
[V2G2-503]	—	—	—	—	—	—	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_Authorization_001 TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011
[V2G2-504]	—	—	—	—	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-504]
[V2G2-505]	—	—	—	—	P	P	P	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-505]
[V2G2-506]	—	—	—	—	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-506]
[V2G2-507]	—	—	—	—	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-507]
[V2G2-508]	—	—	—	—	X	X	X	X	Group_[V2G2-508]
[V2G2-509]	—	—	—	—	P	P	—	—	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_Authorization_002
[V2G2-510]	—	—	—	—	P	—	P	—	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-510]_[V2G2-847]_[V2G2-846]

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-511]	—	—	—	—	P	—	P	—	Only one case can be considered, so the requirement can only be tested partially TC_EVCC_AC_VTB_ChargingStatus_002 TC_EVCC_AC_VTB_ChargingStatus_003 TC_EVCC_AC_VTB_ChargingStatus_004 TC_EVCC_AC_VTB_ChargingStatus_005
[V2G2-512]	—	—	—	—	—	—	P	—	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-512]_[V2G2-408]
[V2G2-513]	—	—	—	—	P	—	P	—	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013
[V2G2-514]	—	—	—	—	P	—	P	—	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_AC_VTB_ChargingStatus_001
[V2G2-516]	—	—	—	—	P	—	P	—	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013
[V2G2-517]	—	—	—	—	—	—	P	—	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-517]
[V2G2-518]	—	—	—	—	—	—	P	—	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014
[V2G2-519]	—	—	—	—	—	—	P	—	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014
[V2G2-520]	—	—	—	—	P	—	P	—	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-520]
[V2G2-521]	—	—	—	—	X	—	X	—	Requirement is only testable if a corresponding PIXIT is considered. TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013
[V2G2-522]	—	—	—	—	—	—	X	—	Requirement is only testable if a corresponding PIXIT is considered. TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014
[V2G2-524]	—	—	—	—	—	P	—	P	Only one case can be considered, so the requirement can only be tested partially TC_EVCC_DC_VTB_CableCheck_002

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_EVCC_DC_VTB_CableCheck_003 TC_EVCC_DC_VTB_CableCheck_004 TC_EVCC_DC_VTB_CableCheck_005
[V2G2-525]	—	—	—	—	—	P	—	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_PreCharge_001
[V2G2-526]	—	—	—	—	—	P	—	P	Only one case can be considered, so the requirement can only be tested partially TC_EVCC_DC_VTB_PreCharge_002 TC_EVCC_DC_VTB_PreCharge_003 TC_EVCC_DC_VTB_PreCharge_004 TC_EVCC_DC_VTB_PreCharge_005
[V2G2-527]	—	—	—	—	—	P	—	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001
[V2G2-528]	—	—	—	—	—	P	—	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group [V2G2-528]
[V2G2-530]	—	—	—	—	—	P	—	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_CurrentDemand_001
[V2G2-531]	—	—	—	—	—	P	—	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013
[V2G2-532]	—	—	—	—	—	P	—	P	Only one case can be considered, so the requirement can only be tested partially TC_EVCC_DC_VTB_CurrentDemand_002 TC_EVCC_DC_VTB_CurrentDemand_003 TC_EVCC_DC_VTB_CurrentDemand_004 TC_EVCC_DC_VTB_CurrentDemand_005
[V2G2-533]	—	—	—	—	—	P	—	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_WeldingDetection_001 TC_EVCC_DC_VTB_WeldingDetection_009
[V2G2-534]	—	—	—	—	—	P	—	P	Requirement is only testable if a corresponding PIXIT is considered. Only one case can be considered, so the requirement can only be tested partially TC_EVCC_DC_VTB_WeldingDetection_002 TC_EVCC_DC_VTB_WeldingDetection_003 TC_EVCC_DC_VTB_WeldingDetection_004 TC_EVCC_DC_VTB_WeldingDetection_005
[V2G2-535]	—	—	—	—	—	P	—	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									partially Group_[V2G2-535]
[V2G2-535]	—	—	—	—	—	P	—	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-535]
[V2G2-536]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-537] TC_SECC_CMN_VTB_SupportedAppProtocol_003
[V2G2-537]	X	X	X	X	—	—	—	—	TC_SECC_CMN_VTB_SupportedAppProtocol_003
[V2G2-538]	P	P	P	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-459]_[V2G2-538]
[V2G2-539]	X	X	X	X	—	—	—	—	Group_[V2G2-539]_[V2G2-736]
[V2G2-541]	P	P	P	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_004 TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-543]	P	P	P	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-190]_[V2G2-191]_[V2G2-543]_[V2G2-192]
[V2G2-545]	P	P	P	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-548]	P	P	P	P	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-551]	P	P	P	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_PaymentServiceSelection_001 TC_SECC_CMN_VTB_PaymentServiceSelection_002 TC_SECC_CMN_VTB_PaymentServiceSelection_003
[V2G2-554]	—	—	P	P	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_CertificateInstallation_001
[V2G2-557]	—	—	P	P	—	—	—	—	Requirement is only testable if a corresponding PIXIT is

Requirement ID	Covered in Test Suite								TC ID(s)/Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_CertificateUpdate_001 TC_SECC_CMN_VTB_CertificateUpdate_002

[V2G2-560]	—	—	P	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_PaymentDetails_001 TC_SECC_CMN_VTB_PaymentDetails_002 TC_SECC_CMN_VTB_PaymentDetails_003
[V2G2-563]	P	P	P	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-563]_[V2G2-856]
[V2G2-568]	P	P	P	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_PowerDelivery_005 TC_SECC_DC_VTB_SessionStop_001
[V2G2-571]	P	P	P	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_SessionStop_001 TC_SECC_DC_VTB_SessionStop_001 TC_SECC_DC_VTB_SessionStop_002
[V2G2-573]	P	—	P	—	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-573]_[V2G2-362]_[V2G2-363]
[V2G2-575]	P	—	P	—	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-575]_[V2G2-577]
[V2G2-576]	P	—	P	—	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_PowerDelivery_001 TC_SECC_AC_VTB_PowerDelivery_002 TC_SECC_AC_VTB_PowerDelivery_004
[V2G2-577]	P	—	P	—	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-575]_[V2G2-577]
[V2G2-580]	—	—	P	—	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_MeteringReceipt_001
[V2G2-582]	—	P	—	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-582]
[V2G2-584]	—	P	—	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration,

									so the requirement can only be tested partially TC_SECC_DC_VTB_CableCheck_001 TC_SECC_DC_VTB_CableCheck_002 TC_SECC_DC_VTB_CableCheck_006 TC_SECC_DC_VTB_CableCheck_007
[V2G2-587]	—	P	—	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_PreCharge_001
[V2G2-590]	—	P	—	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_PowerDelivery_001 TC_SECC_DC_VTB_PowerDelivery_012
[V2G2-593]	—	P	—	P	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-593]
[V2G2-597]	—	P	—	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_WeldingDetection_001
[V2G2-599]	—	—	—	—	—	P	—	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_CableCheck_001
[V2G2-601]	—	P	—	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-601]
[V2G2-602]	P	P	P	P	—	—	—	—	Table 7 can only be tested partially. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-603]	—	—	—	—	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_SupportedAppProtocol_001
[V2G2-605]	P	P	P	P	P	P	P	P	Table 111 can only be tested partially Group_[V2G2-605]
[V2G2-606]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-836]_[V2G2-284]_[V2G2-606]_[V2G2-288]_[V2G2-607]
[V2G2-607]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-836]_[V2G2-284]_[V2G2-606]_[V2G2-288]_[V2G2-607]
[V2G2-608]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-294]_[V2G2-608]_[V2G2-299]_[V2G2-609]_[V2G2-300]
[V2G2-609]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-294]_[V2G2-608]_[V2G2-299]_[V2G2-609]_[V2G2-300]
[V2G2-610]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-611]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column

									Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-612]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_[V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-613]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_[V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-614]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-617]	—	—	—	—	—	P	—	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_CableCheck_006
[V2G2-618]	—	—	—	—	—	P	—	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_PreCharge_006
[V2G2-619]	—	—	—	—	—	P	—	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-619]
[V2G2-620]	—	—	—	—	—	P	—	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_WeldingDetection_009
[V2G2-621]	—	P	—	P	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_CableCheck_006 TC_SECC_DC_VTB_CableCheck_007
[V2G2-623]	—	—	—	—	P	P	P	P	Table 16 can only be tested partially Group_[V2G2-142]_[V2G2-623]_[V2G2-018]_[V2G2-019]
[V2G2-624]	P	P	P	P	—	—	—	—	Table 16 can only be tested partially TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_003 TC_SECC_CMN_VTB_SDP_007
[V2G2-625]	X	X	—	—	—	—	—	—	TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002
[V2G2-626]	X	X	X	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_003 TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_006

[V2G2-627]	X	X	—	—	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered. TC_SECC_CMN_VTB_SDP_007
[V2G2-628]	—	—	—	—	I	I	I	I	Tested indirectly through test for requirement [V2G2-166] TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-630]	—	—	—	—	—	—	X	X	TC_EVCC_CMN_VTB_SDP_004
[V2G2-631]	—	—	X	X	—	—	—	—	TC_SECC_CMN_VTB_SDP_003
[V2G2-632]	X	X	—	—	—	—	—	—	TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-633]	—	—	—	—	I	I	—	—	Requirement is only testable if a corresponding PIXIT is considered. Tested indirectly through test for requirement [V2G2-520], [V2G2-535] or [V2G2-619] Group_[V2G2-633]_[V2G2-635]_[V2G2-637]
[V2G2-634]	X	X	—	—	—	—	—	—	Group_[V2G2-634]
[V2G2-635]	—	—	—	—	I	I	—	—	Requirement is only testable if a corresponding PIXIT is considered. Tested indirectly through test for requirement [V2G2-520], [V2G2-535] or [V2G2-619] Group_[V2G2-633]_[V2G2-635]_[V2G2-637]
[V2G2-636]	X	X	—	—	—	—	—	—	TC_SECC_CMN_VTB_PaymentServiceSelection_001
[V2G2-637]	—	—	—	—	I	I	—	—	Requirement is only testable if a corresponding PIXIT is considered. Tested indirectly through test for requirement [V2G2-520], [V2G2-535] or [V2G2-619] Group_[V2G2-633]_[V2G2-635]_[V2G2-637]
[V2G2-645]	—	—	—	—	I	I	—	I	Tested indirectly through test for requirement [V2G2-166] TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-646]	—	—	—	—	X	X	—	—	Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_SupportedAppProtocol_004
[V2G2-652]	—	—	—	—	—	—	X	X	Group_[V2G2-006]_[V2G2-652]_[V2G2-909]
[V2G2-653]	—	—	P	P	—	—	P	P	The verification of this requirement consists of two different test cases, so the requirement can only be tested partially Group_[V2G2-653]
[V2G2-656]	—	—	X	X	—	—	X	X	Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]
[V2G2-659]	—	—	—	—	P	P	P	P	Table 104 can only be tested partially Group_[V2G2-659]_[V2G2-660]_[V2G2-661]_[V2G2-662]
[V2G2-660]	—	—	—	—	P	P	P	P	Table 104 can only be tested partially Group_[V2G2-659]_[V2G2-660]_[V2G2-661]_[V2G2-662]
[V2G2-661]	—	—	—	—	P	P	P	P	Table 104 can only be tested partially Group_[V2G2-659]_[V2G2-660]_[V2G2-661]_[V2G2-662]
[V2G2-662]	—	—	—	—	P	P	P	P	Table 104 can only be tested partially Group_[V2G2-659]_[V2G2-660]_[V2G2-661]_[V2G2-662]
[V2G2-664]	P	P	P	P	—	—	—	—	Table 104 can only be tested partially Group_[V2G2-664]_[V2G2-666]_[V2G2-667]

[V2G2-665]	P	P	P	P	—	—	—	—	Table 104 can only be tested partially Group_[V2G2-665]
[V2G2-666]	P	P	P	P	—	—	—	—	Table 104 can only be tested partially Group_[V2G2-664]_[V2G2-666]_[V2G2-667]
[V2G2-667]	P	P	P	P	—	—	—	—	Table 104 can only be tested partially Group_[V2G2-664]_[V2G2-666]_[V2G2-667]
[V2G2-672]	X	X	X	X	—	—	—	—	Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-673]	—	—	—	—	X	X	X	X	Group_[V2G2-293]_[V2G2-673]
[V2G2-675]	—	—	—	—	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-675]
[V2G2-677]	P	P	P	P	—	—	—	—	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-677]
[V2G2-679]	—	—	—	—	X	X	X	X	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002
[V2G2-680]	—	—	—	—	X	X	X	X	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009
[V2G2-683]	—	—	—	—	P	P	P	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-683]
[V2G2-684]	—	—	—	—	P	P	P	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_Authorization_009
[V2G2-685]	—	—	—	—	P	P	P	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_AC_VTB_ChargeParameterDiscovery_007 TC_EVCC_DC_VTB_ChargeParameterDiscovery_007
[V2G2-686]	—	—	—	—	—	P	—	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008
[V2G2-687]	P	P	P	P	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_Authorization_014 TC_SECC_CMN_VTB_Authorization_015 TC_SECC_CMN_VTB_Authorization_016 TC_SECC_CMN_VTB_Authorization_017
[V2G2-688]	P	P	P	P	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_ChargeParameterDiscovery_014 TC_SECC_DC_VTB_ChargeParameterDiscovery_014
[V2G2-689]	—	—	—	—	X	—	X	—	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009

[V2G2-691]	X	—	—	—	—	—	—	—	TC_SECC_AC_VTB_ChargingStatus_001 TC_SECC_AC_VTB_ChargingStatus_006 TC_SECC_AC_VTB_ChargingStatus_008
[V2G2-696]	—	—	X	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-696]
[V2G2-700]	—	—	—	—	—	I	—	I	Tested indirectly through test for requirement [V2G2-702] TC_EVCC_DC_VTB_CableCheck_006
[V2G2-701]	—	—	—	—	—	I	—	I	Tested indirectly through test for requirement [V2G2-702] or [V2G2-525] TC_EVCC_DC_VTB_CableCheck_006
[V2G2-702]	—	—	—	—	—	P	—	P	Internal states and timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_CableCheck_006
[V2G2-704]	—	—	—	—	—	I	—	I	Tested indirectly through test for requirement [V2G2-706] TC_EVCC_DC_VTB_PreCharge_006
[V2G2-705]	—	—	—	—	—	I	—	I	Tested indirectly through test for requirement [V2G2-706] or [V2G2-528] Group_[V2G2-705]
[V2G2-706]	—	—	—	—	—	P	—	P	Internal states and timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_PreCharge_006
[V2G2-708]	X	—	X	—	—	—	—	—	Group_[V2G2-931]_[V2G2-708]
[V2G2-710]	—	—	—	—	I	I	I	I	Tested indirectly through test for requirement [V2G2-711] TC_EVCC_CMN_VTB_Authorization_009 TC_EVCC_AC_VTB_ChargeParameterDiscovery_007 TC_EVCC_DC_VTB_ChargeParameterDiscovery_007
[V2G2-711]	—	—	—	—	P	P	P	P	TC_EVCC_CMN_VTB_Authorization_009 TC_EVCC_AC_VTB_ChargeParameterDiscovery_007 TC_EVCC_DC_VTB_ChargeParameterDiscovery_007
[V2G2-712]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-713]. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-712]_[V2G2-713]
[V2G2-713]	P	P	P	P	—	—	—	—	Internal states and timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-712]_[V2G2-713]
[V2G2-714]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-716] TC_SECC_CMN_VTB_SessionSetup_004
[V2G2-715]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-716] TC_SECC_CMN_VTB_SessionSetup_004
[V2G2-716]	P	P	P	P	—	—	—	—	Internal states and timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_SessionSetup_004
[V2G2-717]	—	—	—	—	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-717]
[V2G2-718]	—	—	—	—	P	P	P	P	Internal states cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially. Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-718]_[V2G2-739]
[V2G2-721]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-168]

									TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_004 TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-723]	P	P	P	P	—	—	—	—	Internal states and timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_SDP_004 TC_SECC_CMN_VTB_SDP_005 TC_SECC_CMN_VTB_SDP_006
[V2G2-724]	X	X	X	X	—	—	—	—	TC_SECC_AC_VTB_SessionStop_001 TC_SECC_DC_VTB_SessionStop_001 TC_SECC_DC_VTB_SessionStop_002
[V2G2-725]	P	P	P	P	—	—	—	—	Internal states cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially. (Requirements refers to ISO 15118-5) Group_[V2G2-725]
[V2G2-728]	—	—	—	—	X	X	X	X	Group_[V2G2-728]_[V2G2-025]
[V2G2-729]	X	X	X	X	—	—	—	—	Group_[V2G2-729]_[V2G2-034]
[V2G2-731]	—	—	—	—	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-158]_[V2G2-731]
[V2G2-733]	P	P	P	P	—	—	—	—	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-733]
[V2G2-736]	X	X	X	X	—	—	—	—	Group_[V2G2-539]_[V2G2-736]
[V2G2-737]	—	—	—	—	P	P	P	P	Requirement definition considers various possible scenarios, so the requirement can only be tested partially for exemplarily selected cases Group_[V2G2-737]
[V2G2-738]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-239]_[V2G2-738]
[V2G2-739]	—	—	—	—	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-718]_[V2G2-739]
[V2G2-740]	—	—	—	—	P	P	P	P	Requirement is only testable if a corresponding PIXIT is considered. Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-740]
[V2G2-741]	P	P	P	P	—	—	—	—	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-741]_[V2G2-745]
[V2G2-742]	—	—	—	—	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-742]
[V2G2-744]	—	—	—	—	I	I	I	I	Tested indirectly through test for requirement [V2G2-740]. Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-744]
[V2G2-745]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-741] Group_[V2G2-741]_[V2G2-745]
[V2G2-746]	—	—	—	—	X	X	X	X	TC_EVCC_CMN_VTB_SessionSetup_001 TC_EVCC_CMN_VTB_SessionSetup_008
[V2G2-747]	—	—	—	—	P	P	P	P	Only one message sequence will be considered, so the requirement can only be tested partially Group_[V2G2-747]

[V2G2-748]	—	—	—	—	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-748]
[V2G2-750]	X	X	X	X	—	—	—	—	TC_SECC_CMN_VTB_SessionSetup_001 TC_SECC_AC_VTB_SessionSetup_003 TC_SECC_AC_VTB_SessionSetup_006 TC_SECC_DC_VTB_SessionSetup_003 TC_SECC_DC_VTB_SessionSetup_006
[V2G2-751]	P	P	P	P	X	X	X	X	Only one message sequence will be considered, so the requirement can only be tested partially TC_EVCC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_SessionStop_001 TC_SECC_DC_VTB_SessionStop_001 TC_SECC_DC_VTB_SessionStop_002
[V2G2-752]	P	P	P	P	—	—	—	—	Only one message sequence will be considered, so the requirement can only be tested partially Group_[V2G2-752]
[V2G2-753]	I	I	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-754] TC_SECC_AC_VTB_SessionSetup_001 TC_SECC_AC_VTB_SessionSetup_004 TC_SECC_DC_VTB_SessionSetup_001 TC_SECC_DC_VTB_SessionSetup_004
[V2G2-754]	X	X	X	X	—	—	—	—	TC_SECC_AC_VTB_SessionSetup_001 TC_SECC_AC_VTB_SessionSetup_004 TC_SECC_DC_VTB_SessionSetup_001 TC_SECC_DC_VTB_SessionSetup_004
[V2G2-756]	X	X	X	X	—	—	—	—	TC_SECC_AC_VTB_SessionSetup_002 TC_SECC_AC_VTB_SessionSetup_005 TC_SECC_DC_VTB_SessionSetup_002 TC_SECC_DC_VTB_SessionSetup_005
[V2G2-757]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-758]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-759]	P	P	P	P	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Table 63 can only be tested partially TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-762]	X	X	—	—	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered

									Group_{[V2G2-762]}
[V2G2-764]	—	—	—	—	—	—	X	X	Group_{[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]}
[V2G2-765]	—	—	—	—	—	—	X	X	Group_{[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]}

STANDARDSISO.COM : Click to view the full PDF of ISO 15118-4:2018

[V2G2-766]	—	—	—	—	—	—	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-767]	—	—	—	—	—	—	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-768]	—	—	—	—	—	—	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-769]	—	—	—	—	—	—	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-770]	—	—	—	—	—	—	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-771]	—	—	—	—	—	—	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-773]	X	X	X	X	—	—	—	—	Group_[V2G2-773]
[V2G2-776]	—	—	—	—	—	—	X	X	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-776]
[V2G2-778]	—	—	X	X	—	—	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-779]	—	—	X	X	—	—	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-780]	—	—	X	X	—	—	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-781]	—	—	X	X	—	—	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-782]	—	—	X	X	—	—	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-783]	—	—	X	X	—	—	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-787]	—	—	—	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_DC_VTB_CurrentDemand_002 TC_SECC_DC_VTB_CurrentDemand_007

									TC_SECC_DC_VTB_CurrentDemand_009
[V2G2-788]	—	—	—	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_DC_VTB_CurrentDemand_001 TC_SECC_DC_VTB_CurrentDemand_006 TC_SECC_DC_VTB_CurrentDemand_008
[V2G2-789]	—	—	—	—	—	—	—	X	Group_[V2G2-790]_[V2G2-789]
[V2G2-790]	—	—	—	—	—	—	—	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-790]_[V2G2-789]
[V2G2-791]	—	—	—	—	—	—	—	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-791]
[V2G2-792]	—	—	—	—	—	—	—	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014
[V2G2-793]	—	—	—	—	—	—	—	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002
[V2G2-794]	—	—	—	—	—	—	—	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009
[V2G2-795]	—	—	—	P	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_CurrentDemand_002 TC_SECC_DC_VTB_CurrentDemand_007 TC_SECC_DC_VTB_CurrentDemand_009
[V2G2-797]	—	—	—	P	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_MeteringReceipt_001
[V2G2-799]	—	—	—	—	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-799]
[V2G2-800]	I	I	I	I	I	I	I	I	Tested indirectly through test for requirement [V2G2-160] Group_[V2G2-800]
[V2G2-804]	X	X	X	X	—	—	—	—	TC_SECC_CMN_VTB_PaymentServiceSelection_007
[V2G2-812]	P	P	P	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_PowerDelivery_014 TC_SECC_DC_VTB_PowerDelivery_013
[V2G2-813]	P	P	P	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_PowerDelivery_003 TC_SECC_DC_VTB_PowerDelivery_011
[V2G2-814]	—	—	X	X	—	—	—	—	TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-817]	—	—	—	—	—	—	I	I	Tested indirectly through test for requirement [V2G2-500]/[V2G2-501] and [V2G2-503]

									TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011
[V2G2-818]	—	—	—	—	—	—	I	I	Tested indirectly through test for requirement [V2G2-500]/[V2G2-501] and [V2G2-503] TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011
[V2G2-820]	—	—	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-554] and [V2G2-653] TC_SECC_CMN_VTB_Authorization_012 TC_SECC_CMN_VTB_Authorization_016
[V2G2-821]	—	—	I	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-557] and [V2G2-653] TC_SECC_CMN_VTB_Authorization_013 TC_SECC_CMN_VTB_Authorization_017
[V2G2-822]	—	—	P	P	—	—	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]
[V2G2-827]	—	—	—	—	—	—	X	X	TC_EVCC_CMN_VTB_Authorization_011
[V2G2-828]	—	—	—	—	X	—	—	—	TC_EVCC_AC_VTB_PaymentServiceSelection_001
[V2G2-829]	—	—	—	—	X	—	X	—	Requirement is only testable if a corresponding PIXIT is considered. Group_[V2G2-829]
[V2G2-831]	P	P	P	P	—	—	P	P	Requirement is only testable if a corresponding PIXIT is considered. Table 66 can only be tested partially. Group_[V2G2-831]
[V2G2-832]	P	P	P	P	P	P	P	P	Table 68 can only be tested partially Group_[V2G2-832]
[V2G2-836]	—	—	—	—	X	X	X	X	Group_[V2G2-836]_[V2G2-284]_[V2G2-606]_[V2G2-288]_[V2G2-607]
[V2G2-839]	—	—	—	—	X	X	X	X	Group_[V2G2-839]
[V2G2-840]	—	—	—	—	X	X	X	X	Group_[V2G2-840]_[V2G2-841]
[V2G2-841]	—	—	—	—	I	I	I	I	Requirement is only testable if a corresponding PIXIT is considered. Tested indirectly through test for requirement [V2G2-840] Group_[V2G2-840]_[V2G2-841]
[V2G2-842]	—	—	—	—	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered. Group_[V2G2-842]
[V2G2-845]	—	—	—	—	X	X	—	—	TC_EVCC_CMN_VTB_Authorization_009
[V2G2-846]	—	—	—	—	X	—	X	—	Group_[V2G2-510]_[V2G2-847]_[V2G2-846]
[V2G2-847]	—	—	—	—	X	—	X	—	Group_[V2G2-510]_[V2G2-847]_[V2G2-846]
[V2G2-848]	—	—	—	—	X	—	X	—	Group_[V2G2-848]_[V2G2-849]
[V2G2-849]	—	—	—	—	I	—	I	—	Tested indirectly through test for requirement [V2G2-848] Group_[V2G2-848]_[V2G2-849]
[V2G2-852]	X	—	—	—	—	—	—	—	Group_[V2G2-852]
[V2G2-854]	X	X	—	—	—	—	—	—	Cannot be tested explicitly in SECC black box test configuration TC_SECC_CMN_VTB_Authorization_014
[V2G2-855]	—	—	X	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered. TC_SECC_CMN_VTB_Authorization_015 TC_SECC_CMN_VTB_Authorization_016 TC_SECC_CMN_VTB_Authorization_017
[V2G2-856]	X	X	X	X	—	—	—	—	Group_[V2G2-563]_[V2G2-856]
[V2G2-861]	P	—	P	—	—	—	—	—	The contactor measuring cannot be tested in SECC black box test

									configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_PowerDelivery_001 TC_SECC_AC_VTB_PowerDelivery_002 TC_SECC_AC_VTB_PowerDelivery_003 TC_SECC_AC_VTB_PowerDelivery_004
[V2G2-864]	I	—	I	—	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_PowerDelivery_005
[V2G2-871]	P	P	P	P	—	—	—	—	OCSP is out of scope, so the requirement can only be tested partially for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-875]	—	—	—	—	P	P	P	P	OCSP is out of scope, so the requirement can only be tested partially for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_SupportedAppProtocol_001
[V2G2-879]	—	—	—	—	X	X	X	X	Has not been implemented yet due to a higher complexity TC_EVCC_CMN_VTB_SessionSetup_008
[V2G2-881]	—	P	—	P	—	—	—	—	Table 100 can only be tested partially Group_[V2G2-881]
[V2G2-884]	N	N	P	P	N	N	P	P	Cannot be tested explicitly in SECC/EVCC black box test configuration for the profile defined in the respective column. Only one case can be considered, so the requirement can only be tested partially for the profile defined in the respective column. TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001 TC_EVCC_CMN_VTB_CertificateUpdate_001 TC_EVCC_CMN_VTB_PaymentDetails_001 TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-888]	—	—	—	—	—	—	X	X	TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-891]	—	—	—	—	—	—	P	P	The verification of this requirement consists of two different test cases, so the requirement can only be tested partially Group_[V2G2-891]
[V2G2-893]	—	—	—	—	—	—	X	X	TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-896]	—	—	—	—	—	—	P	P	The verification of this requirement consists of two different test cases, so the requirement can only be tested partially Group_[V2G2-896]
[V2G2-898]	—	—	—	—	—	—	X	X	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001 TC_EVCC_CMN_VTB_PaymentDetails_001 TC_EVCC_CMN_VTB_PaymentDetails_002 TC_EVCC_CMN_VTB_PaymentDetails_003
[V2G2-899]	—	—	P	P	—	—	—	—	The verification of this requirement consists of two different test cases, so the requirement can only be tested partially Group_[V2G2-899]
[V2G2-900]	—	—	—	—	—	—	X	X	TC_EVCC_CMN_VTB_Authorization_001 TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011
[V2G2-901]	—	—	P	P	—	—	—	—	The verification of this requirement consists of two different test cases, so the requirement can only be tested partially

									TC_SECC_CMN_VTB_Authorization_002 TC_SECC_CMN_VTB_Authorization_007 TC_SECC_CMN_VTB_Authorization_010 TC_SECC_CMN_VTB_Authorization_011 TC_SECC_CMN_VTB_Authorization_015
[V2G2-903]	—	—	—	—	—	—	X	X	Group_[V2G2-903]
[V2G2-904]	—	—	I	I	—	—	—	—	Tested indirectly through test for requirements [V2G2-580] or [V2G2-797] TC_SECC_AC_VTB_MeteringReceipt_001 TC_SECC_DC_VTB_MeteringReceipt_001
[V2G2-905]	N	N	I	I	—	—	—	—	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column. Tested indirectly through test of signature validation for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered. Group_[V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_[V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-906]	—	—	X	X	—	—	—	—	Requirement is only testable if a corresponding PIXIT is considered. Group_[V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_[V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-909]	—	—	—	—	—	-	X	X	Group_[V2G2-006]_[V2G2-652]_[V2G2-909]
[V2G2-912]	—	—	—	—	—	X	—	X	TC_EVCC_DC_VTB_CableCheck_001
[V2G2-913]	—	—	—	—	—	X	—	X	Group_[V2G2-913]_[V2G2-914]
[V2G2-914]	—	—	—	—	—	I	—	I	Tested indirectly through test for requirement [V2G2-913] Group_[V2G2-913]_[V2G2-914]
[V2G2-915]	—	P	—	P	—	—	—	—	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-915]
[V2G2-917]	—	P	—	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_CableCheck_001 TC_SECC_DC_VTB_CableCheck_002 TC_SECC_DC_VTB_CableCheck_006 TC_SECC_DC_VTB_CableCheck_007
[V2G2-918]	—	X	—	X	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_CableCheck_008
[V2G2-919]	—	I	—	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-922] TC_SECC_DC_VTB_SessionStop_006 TC_SECC_DC_VTB_WeldingDetection_005
[V2G2-920]	—	I	—	I	—	—	—	—	Tested indirectly through test for requirement [V2G2-922] TC_SECC_DC_VTB_SessionStop_006 TC_SECC_DC_VTB_WeldingDetection_005
[V2G2-921]	—	P	—	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_SessionStop_001 TC_SECC_DC_VTB_SessionStop_002 TC_SECC_DC_VTB_WeldingDetection_001
[V2G2-922]	—	P	—	P	—	—	—	—	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_SessionStop_006 TC_SECC_DC_VTB_WeldingDetection_005
[V2G2-926]	N	N	X	X	N	N	X	X	Cannot be tested explicitly in SECC/EVCC black box test configuration for the

									profile defined in the respective column. Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]
[V2G2-930]	—	—	—	—	I	—	I	—	Tested indirectly through the PTC for IEC 61851-1 signaling (see 7.7.7) Group_[V2G2-930]
[V2G2-931]	I	—	I	—	—	—	—	—	Tested indirectly through the PTC for IEC 61851-1 signaling (see 7.7.7) Group_[V2G2-931]_[V2G2-708]
[V2G2-932]	—	—	—	—	—	—	X	X	TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-933]	—	—	—	—	—	—	P	P	Only one field can be considered, so the requirement can only be tested partially TC_EVCC_CMN_VTB_CertificateInstallation_001

Several requirements of ISO 15118-2 are covered in multiple test cases of the ATS. For a simplified representation, the relevant TC identifier of each test case is mapped to the corresponding requirement. These special groups are summarized in Table 30. If two requirement groups refer to an identical set of TC identifiers, these groups are consolidated.

Table 30 — Groups for a simplified TC Id representation (see Table 29)

Group name	TC IDs
Group_[V2G2-159]	TC_EVCC_CMN_VTB_SDP_005, TC_EVCC_CMN_VTB_SDP_006, TC_EVCC_AC_VTB_SDP_001, TC_EVCC_AC_VTB_SDP_002, TC_EVCC_AC_VTB_SDP_003, TC_EVCC_AC_VTB_SDP_004, TC_EVCC_AC_VTB_SDP_005, TC_EVCC_AC_VTB_SDP_006, TC_EVCC_AC_VTB_SDP_007, TC_EVCC_AC_VTB_SDP_008, TC_EVCC_AC_VTB_SDP_009, TC_EVCC_AC_VTB_SDP_010, TC_EVCC_AC_VTB_SDP_011, TC_EVCC_AC_VTB_SDP_012, TC_EVCC_AC_VTB_SDP_013, TC_EVCC_AC_VTB_SDP_014, TC_EVCC_AC_VTB_SDP_015, TC_EVCC_AC_VTB_SDP_016, TC_EVCC_DC_VTB_SDP_001, TC_EVCC_DC_VTB_SDP_002, TC_EVCC_DC_VTB_SDP_003, TC_EVCC_DC_VTB_SDP_004, TC_EVCC_DC_VTB_SDP_005, TC_EVCC_DC_VTB_SDP_006, TC_EVCC_DC_VTB_SDP_007, TC_EVCC_DC_VTB_SDP_008, TC_EVCC_DC_VTB_SDP_009, TC_EVCC_DC_VTB_SDP_010, TC_EVCC_DC_VTB_SDP_011, TC_EVCC_DC_VTB_SDP_012, TC_EVCC_DC_VTB_SDP_013, TC_EVCC_DC_VTB_SDP_014, TC_EVCC_DC_VTB_SDP_015, TC_EVCC_DC_VTB_SDP_016,
Group_[V2G2-504]	TC_EVCC_CMN_VTB_Authorization_003, TC_EVCC_CMN_VTB_Authorization_004, TC_EVCC_CMN_VTB_Authorization_005, TC_EVCC_CMN_VTB_Authorization_006, TC_EVCC_CMN_VTB_Authorization_007, TC_EVCC_CMN_VTB_Authorization_008,
Group_[V2G2-737]	TC_EVCC_CMN_VTB_Authorization_013, TC_EVCC_CMN_VTB_Authorization_014, TC_EVCC_DC_VTB_CableCheck_008, TC_EVCC_DC_VTB_CableCheck_009, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_007, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_008, TC_EVCC_CMN_VTB_CertificateInstallation_011, TC_EVCC_CMN_VTB_CertificateInstallation_012, TC_EVCC_CMN_VTB_CertificateUpdate_013, TC_EVCC_CMN_VTB_CertificateUpdate_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_009, TC_EVCC_AC_VTB_ChargeParameterDiscovery_010, TC_EVCC_DC_VTB_ChargeParameterDiscovery_009, TC_EVCC_DC_VTB_ChargeParameterDiscovery_010, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_011, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_012, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_011, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_012, TC_EVCC_AC_VTB_ChargingStatus_006, TC_EVCC_AC_VTB_ChargingStatus_007, TC_EVCC_DC_VTB_CurrentDemand_006, TC_EVCC_DC_VTB_CurrentDemand_007, TC_EVCC_AC_VTB_MeteringReceipt_011, TC_EVCC_AC_VTB_MeteringReceipt_012, TC_EVCC_DC_VTB_MeteringReceipt_011, TC_EVCC_DC_VTB_MeteringReceipt_012, TC_EVCC_AC_VTB_PowerDelivery_011, TC_EVCC_AC_VTB_PowerDelivery_012, TC_EVCC_DC_VTB_PreCharge_008, TC_EVCC_DC_VTB_PreCharge_009, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_011, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_012, TC_EVCC_CMN_VTB_ServiceDetail_015, TC_EVCC_CMN_VTB_ServiceDetail_016, TC_EVCC_CMN_VTB_ServiceDetail_017, TC_EVCC_CMN_VTB_ServiceDetail_018,

Group name	TC IDs
	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_011, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_012, TC_EVCC_CMN_VTB_ServiceDiscovery_006, TC_EVCC_CMN_VTB_ServiceDiscovery_007, TC_EVCC_CMN_VTB_SessionSetup_006, TC_EVCC_CMN_VTB_SessionSetup_007, TC_EVCC_DC_VTB_WeldingDetection_007, TC_EVCC_DC_VTB_WeldingDetection_008,
Group_[V2G2-747]	TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_002, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_CMN_VTB_CertificateInstallation_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_AC_VTB_ChargingStatus_001, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013,

Group name	TC IDs
	TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_CMN_VTB_PaymentDetails_003, TC_EVCC_CMN_VTB_PaymentServiceSelection_001, TC_EVCC_CMN_VTB_PaymentServiceSelection_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_003, TC_EVCC_CMN_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_001, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_005, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_009, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_012, TC_EVCC_AC_VTB_PaymentServiceSelection_013, TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_PaymentServiceSelection_015, TC_EVCC_AC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_PaymentServiceSelection_017, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_004, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_007, TC_EVCC_DC_VTB_PaymentServiceSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_011, TC_EVCC_DC_VTB_PaymentServiceSelection_012, TC_EVCC_DC_VTB_PaymentServiceSelection_013, TC_EVCC_DC_VTB_PaymentServiceSelection_014, TC_EVCC_DC_VTB_PaymentServiceSelection_015, TC_EVCC_DC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_CMN_VTB_ServiceDetail_001, TC_EVCC_CMN_VTB_ServiceDetail_002,

Group name	TC IDs
	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_CMN_VTB_ServiceDiscovery_001, TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_008, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_006, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020,
Group_[V2G2-498]	TC_EVCC_CMN_VTB_CertificateInstallation_002, TC_EVCC_CMN_VTB_CertificateInstallation_003, TC_EVCC_CMN_VTB_CertificateInstallation_004, TC_EVCC_CMN_VTB_CertificateInstallation_005, TC_EVCC_CMN_VTB_CertificateInstallation_006, TC_EVCC_CMN_VTB_CertificateInstallation_007, TC_EVCC_CMN_VTB_CertificateInstallation_008, TC_EVCC_CMN_VTB_CertificateInstallation_009,

Group name	TC IDs
Group_[V2G2-499]	TC_EVCC_CMN_VTB_CertificateUpdate_002, TC_EVCC_CMN_VTB_CertificateUpdate_003, TC_EVCC_CMN_VTB_CertificateUpdate_004, TC_EVCC_CMN_VTB_CertificateUpdate_005, TC_EVCC_CMN_VTB_CertificateUpdate_006, TC_EVCC_CMN_VTB_CertificateUpdate_007, TC_EVCC_CMN_VTB_CertificateUpdate_008, TC_EVCC_CMN_VTB_CertificateUpdate_009, TC_EVCC_CMN_VTB_CertificateUpdate_010, TC_EVCC_CMN_VTB_CertificateUpdate_011,
Group_[V2G2-380]	TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_AC_VTB_ChargingStatus_001, TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_CMN_VTB_PaymentDetails_003, TC_EVCC_CMN_VTB_PaymentServiceSelection_001, TC_EVCC_CMN_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_012, TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031,

Group name	TC IDs
	TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_CMN_VTB_ServiceDiscovery_001, TC_EVCC_CMN_VTB_SessionSetup_001, TC_EVCC_CMN_VTB_SessionSetup_008, TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002, TC_EVCC_AC_VTB_SessionSetup_003, TC_EVCC_AC_VTB_SessionSetup_004, TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_CMN_VTB_SupportedAppProtocol_001,
Group_[V2G2-376]	TC_EVCC_CMN_VTB_Authorization_002, TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingStatus_001, TC_EVCC_CMN_VTB_PaymentServiceSelection_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_001, TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_005, TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_009, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_013, TC_EVCC_AC_VTB_PaymentServiceSelection_015, TC_EVCC_AC_VTB_PaymentServiceSelection_017, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_CMN_VTB_ServiceDiscovery_001, TC_EVCC_CMN_VTB_SessionSetup_001, TC_EVCC_CMN_VTB_SessionSetup_008, TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002, TC_EVCC_AC_VTB_SessionSetup_003,

Group name	TC IDs
	TC_EVCC_AC_VTB_SessionSetup_004, TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_008, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020, TC_EVCC_CMN_VTB_SupportedAppProtocol_001, TC_EVCC_CMN_VTB_SupportedAppProtocol_004, TC_EVCC_CMN_VTB_SupportedAppProtocol_005,
Group_[V2G2-832]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreCharge_006, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021,

Group name	TC IDs
	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035,

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_WeldingDetection_001,
Group_[V2G2-505]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026,
Group_[V2G2-930]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026,
Group_[V2G2-215]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014,
Group_[V2G2-740]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017,

Group name	TC IDs
	TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_PaymentServiceSelection_015, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_013, TC_EVCC_DC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002, TC_EVCC_AC_VTB_SessionSetup_003, TC_EVCC_AC_VTB_SessionSetup_004, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002, TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004,
Group_[V2G2-744]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023,

Group name	TC IDs
	TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_005, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_009, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_012, TC_EVCC_AC_VTB_PaymentServiceSelection_013, TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_PaymentServiceSelection_015, TC_EVCC_AC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_PaymentServiceSelection_017, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_004, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_007, TC_EVCC_DC_VTB_PaymentServiceSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_011, TC_EVCC_DC_VTB_PaymentServiceSelection_012, TC_EVCC_DC_VTB_PaymentServiceSelection_013, TC_EVCC_DC_VTB_PaymentServiceSelection_014, TC_EVCC_DC_VTB_PaymentServiceSelection_015, TC_EVCC_DC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002, TC_EVCC_AC_VTB_SessionSetup_003, TC_EVCC_AC_VTB_SessionSetup_004, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002,

Group name	TC IDs
	TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004,
Group_[V2G2-394]	TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_CMN_VTB_PaymentDetails_003, TC_EVCC_CMN_VTB_PaymentServiceSelection_001, TC_EVCC_CMN_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_007, TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_011, TC_EVCC_DC_VTB_PaymentServiceSelection_013, TC_EVCC_DC_VTB_PaymentServiceSelection_015, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_CMN_VTB_ServiceDiscovery_001, TC_EVCC_CMN_VTB_SessionSetup_001, TC_EVCC_CMN_VTB_SessionSetup_008, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002, TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004, TC_EVCC_DC_VTB_SessionStop_001,

Group name	TC IDs
	TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_CMN_VTB_SupportedAppProtocol_001, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018,
Group_[V2G2-506]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_002, TC_EVCC_AC_VTB_ChargeParameterDiscovery_003, TC_EVCC_AC_VTB_ChargeParameterDiscovery_004, TC_EVCC_AC_VTB_ChargeParameterDiscovery_005, TC_EVCC_AC_VTB_ChargeParameterDiscovery_006, TC_EVCC_DC_VTB_ChargeParameterDiscovery_002, TC_EVCC_DC_VTB_ChargeParameterDiscovery_003, TC_EVCC_DC_VTB_ChargeParameterDiscovery_004, TC_EVCC_DC_VTB_ChargeParameterDiscovery_005, TC_EVCC_DC_VTB_ChargeParameterDiscovery_006, TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001,
Group_[V2G2-683]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014,
Group_[V2G2-390]	TC_EVCC_CMN_VTB_Authorization_002, TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_CMN_VTB_PaymentServiceSelection_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_004, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_004, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_012, TC_EVCC_DC_VTB_PaymentServiceSelection_014, TC_EVCC_DC_VTB_PaymentServiceSelection_016, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001,

Group name	TC IDs
	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_CMN_VTB_ServiceDiscovery_001, TC_EVCC_CMN_VTB_SessionSetup_001, TC_EVCC_CMN_VTB_SessionSetup_008, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002, TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_006, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030, TC_EVCC_CMN_VTB_SupportedAppProtocol_001, TC_EVCC_CMN_VTB_SupportedAppProtocol_004, TC_EVCC_CMN_VTB_SupportedAppProtocol_005, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020,
Group_[V2G2-742]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026,
Group_[V2G2-482]	TC_EVCC_CMN_VTB_Authorization_012, TC_EVCC_DC_VTB_CableCheck_007, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_006, TC_EVCC_CMN_VTB_CertificateInstallation_010, TC_EVCC_CMN_VTB_CertificateUpdate_012, TC_EVCC_CMN_VTB_ChargeParameterDiscovery_002,

Group name	TC IDs
	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_010, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_010, TC_EVCC_AC_VTB_ChargingStatus_008, TC_EVCC_DC_VTB_CurrentDemand_008, TC_EVCC_AC_VTB_MeteringReceipt_010, TC_EVCC_DC_VTB_MeteringReceipt_010, TC_EVCC_AC_VTB_PowerDelivery_010, TC_EVCC_DC_VTB_PreCharge_007, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_010, TC_EVCC_CMN_VTB_ServiceDetail_013, TC_EVCC_CMN_VTB_ServiceDetail_014, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_010, TC_EVCC_CMN_VTB_ServiceDiscovery_008, TC_EVCC_CMN_VTB_SessionSetup_005, TC_EVCC_AC_VTB_SessionStop_006, TC_EVCC_DC_VTB_WeldingDetection_006, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_006,
Group_[V2G2-182]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_PowerDelivery_006, TC_SECC_AC_VTB_PowerDelivery_007, TC_SECC_AC_VTB_PowerDelivery_008, TC_SECC_AC_VTB_PowerDelivery_009, TC_SECC_AC_VTB_PowerDelivery_010, TC_SECC_AC_VTB_PowerDelivery_011, TC_SECC_AC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_003, TC_SECC_DC_VTB_PowerDelivery_004, TC_SECC_DC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_006, TC_SECC_DC_VTB_PowerDelivery_007, TC_SECC_DC_VTB_PowerDelivery_008, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021,
Group_[V2G2-839]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009,

Group name	TC IDs
	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002,
Group_[V2G2-242]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_AC_VTB_ChargingStatus_001,
Group_[V2G2-404]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_AC_VTB_ChargingStatus_001, TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_018,
Group_[V2G2-799]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_003, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_004, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_005, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_006, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_003, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_004, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_005, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_006, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007, TC_EVCC_AC_VTB_PowerDelivery_002, TC_EVCC_AC_VTB_PowerDelivery_003, TC_EVCC_AC_VTB_PowerDelivery_004, TC_EVCC_AC_VTB_PowerDelivery_005, TC_EVCC_AC_VTB_PowerDelivery_006, TC_EVCC_AC_VTB_PowerDelivery_007, TC_EVCC_AC_VTB_PowerDelivery_008, TC_EVCC_AC_VTB_PowerDelivery_009, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_002, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_003, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_004, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_005, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_006, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_007, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_008, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009,
Group_[V2G2-405]	TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009,

Group name	TC IDs
	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018,
Group_[V2G2-517]	TC_EVCC_AC_VTB_MeteringReceipt_002, TC_EVCC_AC_VTB_MeteringReceipt_003, TC_EVCC_AC_VTB_MeteringReceipt_004, TC_EVCC_AC_VTB_MeteringReceipt_005, TC_EVCC_AC_VTB_MeteringReceipt_006, TC_EVCC_AC_VTB_MeteringReceipt_007,
Group_[V2G2-903]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021,
Group_[V2G2-831]	TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_009,
Group_[V2G2-776]	TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021,
Group_[V2G2-791]	TC_EVCC_DC_VTB_MeteringReceipt_002, TC_EVCC_DC_VTB_MeteringReceipt_003,

Group name	TC IDs
	TC_EVCC_DC_VTB_MeteringReceipt_004, TC_EVCC_DC_VTB_MeteringReceipt_005, TC_EVCC_DC_VTB_MeteringReceipt_006, TC_EVCC_DC_VTB_MeteringReceipt_007,
Group_[V2G2-891]	TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_CMN_VTB_PaymentDetails_002,
Group_[V2G2-896]	TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_EVCC_CMN_VTB_PaymentDetails_001,
Group_[V2G2-490]	TC_EVCC_CMN_VTB_PaymentServiceSelection_003, TC_EVCC_CMN_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_001, TC_EVCC_AC_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_005, TC_EVCC_AC_VTB_PaymentServiceSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_009, TC_EVCC_AC_VTB_PaymentServiceSelection_012, TC_EVCC_AC_VTB_PaymentServiceSelection_013, TC_EVCC_AC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_PaymentServiceSelection_017, TC_EVCC_DC_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_004, TC_EVCC_DC_VTB_PaymentServiceSelection_007, TC_EVCC_DC_VTB_PaymentServiceSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_011, TC_EVCC_DC_VTB_PaymentServiceSelection_012, TC_EVCC_DC_VTB_PaymentServiceSelection_015, TC_EVCC_DC_VTB_PaymentServiceSelection_016,
Group_[V2G2-402]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingStatus_001, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_008, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020,
Group_[V2G2-675]	TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018,

Group name	TC IDs
	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022,
Group_[V2G2-842]	TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022,
Group_[V2G2-829]	TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033,
Group_[V2G2-296]	TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018,
Group_[V2G2-403]	TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_006, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009,

Group name	TC IDs
	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020,
Group_[V2G2-705]	TC_EVCC_DC_VTB_PreCharge_006, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022,
Group_[V2G2-528]	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022,
Group_[V2G2-491]	TC_EVCC_CMN_VTB_ServiceDetail_003, TC_EVCC_CMN_VTB_ServiceDetail_004, TC_EVCC_CMN_VTB_ServiceDetail_005, TC_EVCC_CMN_VTB_ServiceDetail_006, TC_EVCC_CMN_VTB_ServiceDetail_007, TC_EVCC_CMN_VTB_ServiceDetail_008, TC_EVCC_CMN_VTB_ServiceDetail_009, TC_EVCC_CMN_VTB_ServiceDetail_010, TC_EVCC_CMN_VTB_ServiceDetail_011, TC_EVCC_CMN_VTB_ServiceDetail_012,
Group_[V2G2-492]	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009,
Group_[V2G2-748]	TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002, TC_EVCC_AC_VTB_SessionSetup_003, TC_EVCC_AC_VTB_SessionSetup_004, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002, TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004,
Group_[V2G2-520]	TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_008, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020,
Group_[V2G2-508]	TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_007,

Group name	TC IDs
	TC_EVCC_AC_VTB_SessionStop_008, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_029,
Group_[V2G2-717]	TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011,
Group_[V2G2-535]	TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020,
Group_[V2G2-619]	TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_006, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_030,
Group_[V2G2-507]	TC_EVCC_AC_VTB_SessionStop_002, TC_EVCC_AC_VTB_SessionStop_003, TC_EVCC_AC_VTB_SessionStop_004, TC_EVCC_AC_VTB_SessionStop_005, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_002, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_003, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_004, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005,
Group_[V2G2-800]	TC_EVCC_CMN_VTB_V2GTPSDP_002, TC_EVCC_CMN_VTB_V2GTPSDP_003, TC_EVCC_CMN_VTB_V2GTPSDP_004, TC_EVCC_CMN_VTB_V2GTPSDP_005, TC_EVCC_CMN_VTB_V2GTPSessionSetup_002, TC_EVCC_CMN_VTB_V2GTPSessionSetup_003,

Group name	TC IDs
	TC_EVCC_CMN_VTB_V2GTPSessionSetup_004, TC_EVCC_CMN_VTB_V2GTPSessionSetup_005, TC_SECC_CMN_VTB_V2GTPSDP_002, TC_SECC_CMN_VTB_V2GTPSDP_003, TC_SECC_CMN_VTB_V2GTPSDP_004, TC_SECC_CMN_VTB_V2GTPSDP_005, TC_SECC_CMN_VTB_V2GTPSessionSetup_002, TC_SECC_CMN_VTB_V2GTPSessionSetup_003, TC_SECC_CMN_VTB_V2GTPSessionSetup_004, TC_SECC_CMN_VTB_V2GTPSessionSetup_005,
Group_[V2G2-605]	TC_EVCC_CMN_VTB_Authorization_009, TC_EVCC_DC_VTB_CableCheck_006, TC_EVCC_AC_VTB_ChargeParameterDiscovery_007, TC_EVCC_DC_VTB_ChargeParameterDiscovery_007, TC_EVCC_DC_VTB_PreCharge_006, TC_SECC_CMN_VTB_SDP_004, TC_SECC_CMN_VTB_SDP_005, TC_SECC_CMN_VTB_SDP_006, TC_SECC_CMN_VTB_SessionSetup_004, TC_SECC_CMN_VTB_SupportedAppProtocol_005,
Group_[V2G2-915]	TC_SECC_CMN_VTB_SDP_001, TC_SECC_CMN_VTB_SDP_002, TC_SECC_CMN_VTB_SDP_003, TC_SECC_CMN_VTB_SDP_007, TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_MeteringReceipt_001, TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002,

Group name	TC IDs
	TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_ServiceDetail_012, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_004, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_DC_VTB_WeldingDetection_001,
Group_[V2G2-752]	TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_003, TC_SECC_CMN_VTB_Authorization_004, TC_SECC_CMN_VTB_Authorization_005, TC_SECC_CMN_VTB_Authorization_006, TC_SECC_CMN_VTB_Authorization_007, TC_SECC_CMN_VTB_Authorization_010, TC_SECC_CMN_VTB_Authorization_011, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_003, TC_SECC_DC_VTB_CableCheck_004, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_CableCheck_008, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateInstallation_002, TC_SECC_CMN_VTB_CertificateInstallation_005, TC_SECC_CMN_VTB_CertificateInstallation_006, TC_SECC_CMN_VTB_CertificateInstallation_007, TC_SECC_CMN_VTB_CertificateInstallation_008, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_CMN_VTB_CertificateUpdate_002, TC_SECC_CMN_VTB_CertificateUpdate_003, TC_SECC_CMN_VTB_CertificateUpdate_005, TC_SECC_CMN_VTB_CertificateUpdate_006, TC_SECC_CMN_VTB_CertificateUpdate_007, TC_SECC_CMN_VTB_CertificateUpdate_008, TC_SECC_CMN_VTB_CertificateUpdate_009, TC_SECC_CMN_VTB_CertificateUpdate_010, TC_SECC_CMN_VTB_CertificateUpdate_011, TC_SECC_CMN_VTB_CertificateUpdate_012, TC_SECC_CMN_VTB_CertificateUpdate_013, TC_SECC_CMN_VTB_CertificateUpdate_014, TC_SECC_CMN_VTB_CertificateUpdate_015, TC_SECC_CMN_VTB_CertificateUpdate_016, TC_SECC_CMN_VTB_CertificateUpdate_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_004, TC_SECC_AC_VTB_ChargeParameterDiscovery_005, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029,

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_003, TC_SECC_DC_VTB_ChargeParameterDiscovery_004, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_CMN_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_003, TC_SECC_AC_VTB_ChargingStatus_004, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_003, TC_SECC_DC_VTB_CurrentDemand_004, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_AC_VTB_MeteringReceipt_002, TC_SECC_AC_VTB_MeteringReceipt_003, TC_SECC_AC_VTB_MeteringReceipt_004, TC_SECC_AC_VTB_MeteringReceipt_006, TC_SECC_AC_VTB_MeteringReceipt_007, TC_SECC_DC_VTB_MeteringReceipt_001, TC_SECC_DC_VTB_MeteringReceipt_002, TC_SECC_DC_VTB_MeteringReceipt_003, TC_SECC_DC_VTB_MeteringReceipt_004, TC_SECC_DC_VTB_MeteringReceipt_006, TC_SECC_DC_VTB_MeteringReceipt_007, TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentDetails_004, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_CMN_VTB_PaymentServiceSelection_004, TC_SECC_CMN_VTB_PaymentServiceSelection_006, TC_SECC_CMN_VTB_PaymentServiceSelection_007, TC_SECC_CMN_VTB_PaymentServiceSelection_009,

Group name	TC IDs
	TC_SECC_CMN_VTB_PaymentServiceSelection_010, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_PowerDelivery_006, TC_SECC_AC_VTB_PowerDelivery_007, TC_SECC_AC_VTB_PowerDelivery_008, TC_SECC_AC_VTB_PowerDelivery_009, TC_SECC_AC_VTB_PowerDelivery_010, TC_SECC_AC_VTB_PowerDelivery_011, TC_SECC_AC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_003, TC_SECC_DC_VTB_PowerDelivery_004, TC_SECC_DC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_006, TC_SECC_DC_VTB_PowerDelivery_007, TC_SECC_DC_VTB_PowerDelivery_008, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_PreCharge_002, TC_SECC_DC_VTB_PreCharge_003, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_003, TC_SECC_CMN_VTB_ServiceDetail_004, TC_SECC_CMN_VTB_ServiceDetail_007, TC_SECC_CMN_VTB_ServiceDetail_008, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_ServiceDetail_012, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_AC_VTB_SessionStop_002, TC_SECC_AC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_DC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_004, TC_SECC_DC_VTB_SessionStop_006, TC_SECC_DC_VTB_SessionStop_007, TC_SECC_DC_VTB_SessionStop_008, TC_SECC_DC_VTB_WeldingDetection_001, TC_SECC_DC_VTB_WeldingDetection_002, TC_SECC_DC_VTB_WeldingDetection_003, TC_SECC_DC_VTB_WeldingDetection_005,
Group_[V2G2-665]	TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013,

TC_SECC_DC_VTB_ChargeParameterDiscovery_018,
TC_SECC_DC_VTB_ChargeParameterDiscovery_019,
TC_SECC_DC_VTB_ChargeParameterDiscovery_019,
TC_SECC_DC_VTB_ChargeParameterDiscovery_020,
TC_SECC_DC_VTB_ChargeParameterDiscovery_020,
TC_SECC_DC_VTB_ChargeParameterDiscovery_021,
TC_SECC_DC_VTB_ChargeParameterDiscovery_021,
TC_SECC_DC_VTB_ChargeParameterDiscovery_022,
TC_SECC_DC_VTB_ChargeParameterDiscovery_022,
TC_SECC_DC_VTB_ChargeParameterDiscovery_023,
TC_SECC_DC_VTB_ChargeParameterDiscovery_023,
TC_SECC_DC_VTB_ChargeParameterDiscovery_024,
TC_SECC_DC_VTB_ChargeParameterDiscovery_024,
TC_SECC_DC_VTB_ChargeParameterDiscovery_025,
TC_SECC_DC_VTB_ChargeParameterDiscovery_025,
TC_SECC_DC_VTB_ChargeParameterDiscovery_026,
TC_SECC_DC_VTB_ChargeParameterDiscovery_026,
TC_SECC_DC_VTB_ChargeParameterDiscovery_027,
TC_SECC_DC_VTB_ChargeParameterDiscovery_027,
TC_SECC_DC_VTB_ChargeParameterDiscovery_028,
TC_SECC_DC_VTB_ChargeParameterDiscovery_028,
TC_SECC_DC_VTB_ChargeParameterDiscovery_029,
TC_SECC_DC_VTB_ChargeParameterDiscovery_029,
TC_SECC_DC_VTB_ChargeParameterDiscovery_030,
TC_SECC_DC_VTB_ChargeParameterDiscovery_030,
TC_SECC_DC_VTB_ChargeParameterDiscovery_031,
TC_SECC_DC_VTB_ChargeParameterDiscovery_031,
TC_SECC_DC_VTB_ChargeParameterDiscovery_032,
TC_SECC_DC_VTB_ChargeParameterDiscovery_032,
TC_SECC_DC_VTB_ChargeParameterDiscovery_033,
TC_SECC_DC_VTB_ChargeParameterDiscovery_033,
TC_SECC_DC_VTB_ChargeParameterDiscovery_034,
TC_SECC_DC_VTB_ChargeParameterDiscovery_034,
TC_SECC_DC_VTB_ChargeParameterDiscovery_035,
TC_SECC_DC_VTB_ChargeParameterDiscovery_035,
TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_001,
TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_002,
TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_006,

STANDARDSISO.COM

15118-4:2018

Group name	TC IDs
	TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_ServiceDetail_012, TC_SECC_CMN_VTB_ServiceDetail_012, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_004, TC_SECC_CMN_VTB_SupportedAppProtocol_004, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_DC_VTB_WeldingDetection_001, TC_SECC_DC_VTB_WeldingDetection_001,
Group [V2G2-457]	TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019,

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_DC_VTB_MeteringReceipt_001, TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_011,

Group name	TC IDs
	TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_ServiceDetail_012, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_WeldingDetection_001,
Group_[V2G2-653]	TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_DC_VTB_MeteringReceipt_001,
Group_[V2G2-381]	TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002,

Group name	TC IDs
	TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionSetup_004, TC_SECC_AC_VTB_SessionSetup_005, TC_SECC_AC_VTB_SessionSetup_006, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_006,
Group_[V2G2-395]	TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_MeteringReceipt_001, TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_002, TC_SECC_DC_VTB_SessionSetup_003, TC_SECC_DC_VTB_SessionSetup_004, TC_SECC_DC_VTB_SessionSetup_005, TC_SECC_DC_VTB_SessionSetup_006, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_006,

Group name	TC IDs
	TC_SECC_DC_VTB_WeldingDetection_001,
Group_[V2G2-435]	TC_SECC_CMN_VTB_Authorization_008, TC_SECC_CMN_VTB_Authorization_009, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_DC_VTB_CableCheck_005, TC_SECC_CMN_VTB_CertificateInstallation_004, TC_SECC_CMN_VTB_CertificateUpdate_004, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_CMN_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargingStatus_005, TC_SECC_DC_VTB_CurrentDemand_005, TC_SECC_AC_VTB_MeteringReceipt_005, TC_SECC_DC_VTB_MeteringReceipt_005, TC_SECC_CMN_VTB_PaymentDetails_006, TC_SECC_CMN_VTB_PaymentServiceSelection_008, TC_SECC_AC_VTB_PowerDelivery_012, TC_SECC_AC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_PowerDelivery_009, TC_SECC_DC_VTB_PowerDelivery_010, TC_SECC_DC_VTB_PreCharge_004, TC_SECC_CMN_VTB_ServiceDetail_009, TC_SECC_CMN_VTB_ServiceDetail_010, TC_SECC_CMN_VTB_ServiceDiscovery_004, TC_SECC_CMN_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionStop_004, TC_SECC_DC_VTB_SessionStop_005, TC_SECC_DC_VTB_SessionStop_009, TC_SECC_CMN_VTB_SupportedAppProtocol_003, TC_SECC_DC_VTB_WeldingDetection_004,
Group_[V2G2-391]	TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017,

Group name	TC IDs
	TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_002, TC_SECC_DC_VTB_SessionSetup_003, TC_SECC_DC_VTB_SessionSetup_004, TC_SECC_DC_VTB_SessionSetup_005, TC_SECC_DC_VTB_SessionSetup_006, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_004, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_DC_VTB_WeldingDetection_001,
Group_[V2G2-461]	TC_SECC_CMN_VTB_Authorization_010, TC_SECC_CMN_VTB_Authorization_011, TC_SECC_CMN_VTB_CertificateInstallation_005, TC_SECC_CMN_VTB_CertificateInstallation_006, TC_SECC_CMN_VTB_CertificateUpdate_005, TC_SECC_CMN_VTB_CertificateUpdate_006,
Group_[V2G2-696]	TC_SECC_CMN_VTB_CertificateUpdate_003, TC_SECC_CMN_VTB_CertificateUpdate_005, TC_SECC_CMN_VTB_CertificateUpdate_006, TC_SECC_CMN_VTB_CertificateUpdate_007, TC_SECC_CMN_VTB_CertificateUpdate_008, TC_SECC_CMN_VTB_CertificateUpdate_009, TC_SECC_CMN_VTB_CertificateUpdate_010, TC_SECC_CMN_VTB_CertificateUpdate_011, TC_SECC_CMN_VTB_CertificateUpdate_012, TC_SECC_CMN_VTB_CertificateUpdate_013, TC_SECC_CMN_VTB_CertificateUpdate_014, TC_SECC_CMN_VTB_CertificateUpdate_015, TC_SECC_CMN_VTB_CertificateUpdate_016, TC_SECC_CMN_VTB_CertificateUpdate_017,
Group_[V2G2-458]	TC_SECC_CMN_VTB_CertificateInstallation_008, TC_SECC_CMN_VTB_CertificateUpdate_012, TC_SECC_CMN_VTB_CertificateUpdate_013, TC_SECC_CMN_VTB_CertificateUpdate_014, TC_SECC_CMN_VTB_CertificateUpdate_015, TC_SECC_CMN_VTB_CertificateUpdate_017, TC_SECC_CMN_VTB_PaymentDetails_008, TC_SECC_CMN_VTB_PaymentDetails_009, TC_SECC_CMN_VTB_PaymentDetails_010, TC_SECC_AC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_SessionStop_006, TC_SECC_DC_VTB_WeldingDetection_005,
Group_[V2G2-677]	TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035,

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_WeldingDetection_001,
Group [V2G2-460]	TC_SECC_CMN_VTB_Authorization_005, TC_SECC_CMN_VTB_Authorization_006, TC_SECC_DC_VTB_CableCheck_004, TC_SECC_CMN_VTB_CertificateInstallation_003, TC_SECC_CMN_VTB_CertificateUpdate_003, TC_SECC_AC_VTB_ChargeParameterDiscovery_003, TC_SECC_DC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargingStatus_004, TC_SECC_DC_VTB_CurrentDemand_004, TC_SECC_AC_VTB_MeteringReceipt_003, TC_SECC_DC_VTB_MeteringReceipt_003, TC_SECC_CMN_VTB_PaymentDetails_005, TC_SECC_CMN_VTB_PaymentServiceSelection_005, TC_SECC_AC_VTB_PowerDelivery_008, TC_SECC_AC_VTB_PowerDelivery_009, TC_SECC_DC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_006, TC_SECC_DC_VTB_PreCharge_003, TC_SECC_CMN_VTB_ServiceDetail_005, TC_SECC_CMN_VTB_ServiceDetail_006, TC_SECC_CMN_VTB_ServiceDiscovery_003, TC_SECC_AC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_004, TC_SECC_DC_VTB_SessionStop_008, TC_SECC_DC_VTB_WeldingDetection_003,
Group [V2G2-377]	TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017,

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionSetup_004, TC_SECC_AC_VTB_SessionSetup_005, TC_SECC_AC_VTB_SessionSetup_006, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_004, TC_SECC_CMN_VTB_SupportedAppProtocol_006,
Group_[V2G2-304]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_020,

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-773]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-303]	TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_035,

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-725]	TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_AC_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionSetup_004, TC_SECC_AC_VTB_SessionSetup_005, TC_SECC_AC_VTB_SessionSetup_006, TC_SECC_DC_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_002, TC_SECC_DC_VTB_SessionSetup_003, TC_SECC_DC_VTB_SessionSetup_004, TC_SECC_DC_VTB_SessionSetup_005, TC_SECC_DC_VTB_SessionSetup_006,
Group_[V2G2-582]	TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011,

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-593]	TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_009,
Group_[V2G2-481]	TC_SECC_AC_VTB_MeteringReceipt_004, TC_SECC_AC_VTB_MeteringReceipt_006, TC_SECC_AC_VTB_MeteringReceipt_007, TC_SECC_DC_VTB_MeteringReceipt_004, TC_SECC_DC_VTB_MeteringReceipt_006, TC_SECC_DC_VTB_MeteringReceipt_007,
Group_[V2G2-899]	TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentDetails_007, TC_SECC_CMN_VTB_PaymentDetails_008, TC_SECC_CMN_VTB_PaymentDetails_009, TC_SECC_CMN_VTB_PaymentDetails_010,
Group_[V2G2-601]	TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021,
Group_[V2G2-881]	TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021,
Group_[V2G2-733]	TC_SECC_AC_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionSetup_004, TC_SECC_AC_VTB_SessionSetup_005, TC_SECC_AC_VTB_SessionSetup_006, TC_SECC_DC_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_002, TC_SECC_DC_VTB_SessionSetup_003, TC_SECC_DC_VTB_SessionSetup_004, TC_SECC_DC_VTB_SessionSetup_005, TC_SECC_DC_VTB_SessionSetup_006,
Group_[V2G2-462]	TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionSetup_005, TC_SECC_AC_VTB_SessionSetup_006, TC_SECC_DC_VTB_SessionSetup_002, TC_SECC_DC_VTB_SessionSetup_003, TC_SECC_DC_VTB_SessionSetup_005, TC_SECC_DC_VTB_SessionSetup_006,
Group_[V2G2-634]	TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_007,

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_002,

Group name	TC IDs
	TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_004, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_DC_VTB_WeldingDetection_001,
Group_{[V2G2-762]}	TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_015,
Group_{[V2G2-852]}	TC_SECC_CMN_VTB_SDP_001, TC_SECC_CMN_VTB_SDP_002, TC_SECC_CMN_VTB_SDP_007, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_004, TC_SECC_CMN_VTB_SupportedAppProtocol_006,
Group_{[V2G2-158]}_{[V2G2-731]}	TC_EVCC_AC_VTB_SDP_001, TC_EVCC_AC_VTB_SDP_002, TC_EVCC_AC_VTB_SDP_003, TC_EVCC_AC_VTB_SDP_004, TC_EVCC_AC_VTB_SDP_005, TC_EVCC_AC_VTB_SDP_006, TC_EVCC_AC_VTB_SDP_007, TC_EVCC_AC_VTB_SDP_008, TC_EVCC_AC_VTB_SDP_009, TC_EVCC_AC_VTB_SDP_010, TC_EVCC_AC_VTB_SDP_011, TC_EVCC_AC_VTB_SDP_012, TC_EVCC_AC_VTB_SDP_013, TC_EVCC_AC_VTB_SDP_014, TC_EVCC_AC_VTB_SDP_015, TC_EVCC_AC_VTB_SDP_016, TC_EVCC_DC_VTB_SDP_001, TC_EVCC_DC_VTB_SDP_002, TC_EVCC_DC_VTB_SDP_003, TC_EVCC_DC_VTB_SDP_004, TC_EVCC_DC_VTB_SDP_005, TC_EVCC_DC_VTB_SDP_006, TC_EVCC_DC_VTB_SDP_007, TC_EVCC_DC_VTB_SDP_008, TC_EVCC_DC_VTB_SDP_009, TC_EVCC_DC_VTB_SDP_010, TC_EVCC_DC_VTB_SDP_011, TC_EVCC_DC_VTB_SDP_012, TC_EVCC_DC_VTB_SDP_013, TC_EVCC_DC_VTB_SDP_014, TC_EVCC_DC_VTB_SDP_015, TC_EVCC_DC_VTB_SDP_016,
Group_{[V2G2-728]}_{[V2G2-025]}	TC_EVCC_CMN_VTB_Authorization_003, TC_EVCC_CMN_VTB_Authorization_004, TC_EVCC_CMN_VTB_Authorization_005, TC_EVCC_CMN_VTB_Authorization_006, TC_EVCC_CMN_VTB_Authorization_007, TC_EVCC_CMN_VTB_Authorization_008, TC_EVCC_CMN_VTB_Authorization_009, TC_EVCC_CMN_VTB_Authorization_012, TC_EVCC_CMN_VTB_Authorization_013, TC_EVCC_CMN_VTB_Authorization_014, TC_EVCC_DC_VTB_CableCheck_002, TC_EVCC_DC_VTB_CableCheck_003, TC_EVCC_DC_VTB_CableCheck_004, TC_EVCC_DC_VTB_CableCheck_005, TC_EVCC_DC_VTB_CableCheck_006, TC_EVCC_DC_VTB_CableCheck_007, TC_EVCC_DC_VTB_CableCheck_008, TC_EVCC_DC_VTB_CableCheck_009, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_002, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_003, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_004, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_006, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_007, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_008, TC_EVCC_CMN_VTB_CertificateInstallation_002, TC_EVCC_CMN_VTB_CertificateInstallation_003, TC_EVCC_CMN_VTB_CertificateInstallation_004, TC_EVCC_CMN_VTB_CertificateInstallation_005,

Group name	TC IDs
	TC_EVCC_CMN_VTB_CertificateInstallation_006, TC_EVCC_CMN_VTB_CertificateInstallation_007, TC_EVCC_CMN_VTB_CertificateInstallation_008, TC_EVCC_CMN_VTB_CertificateInstallation_009, TC_EVCC_CMN_VTB_CertificateInstallation_010, TC_EVCC_CMN_VTB_CertificateInstallation_011, TC_EVCC_CMN_VTB_CertificateInstallation_012, TC_EVCC_CMN_VTB_CertificateUpdate_002, TC_EVCC_CMN_VTB_CertificateUpdate_003, TC_EVCC_CMN_VTB_CertificateUpdate_004, TC_EVCC_CMN_VTB_CertificateUpdate_005, TC_EVCC_CMN_VTB_CertificateUpdate_006, TC_EVCC_CMN_VTB_CertificateUpdate_007, TC_EVCC_CMN_VTB_CertificateUpdate_008, TC_EVCC_CMN_VTB_CertificateUpdate_009, TC_EVCC_CMN_VTB_CertificateUpdate_010, TC_EVCC_CMN_VTB_CertificateUpdate_011, TC_EVCC_CMN_VTB_CertificateUpdate_012, TC_EVCC_CMN_VTB_CertificateUpdate_013, TC_EVCC_CMN_VTB_CertificateUpdate_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_002, TC_EVCC_AC_VTB_ChargeParameterDiscovery_003, TC_EVCC_AC_VTB_ChargeParameterDiscovery_004, TC_EVCC_AC_VTB_ChargeParameterDiscovery_005, TC_EVCC_AC_VTB_ChargeParameterDiscovery_006, TC_EVCC_AC_VTB_ChargeParameterDiscovery_007, TC_EVCC_AC_VTB_ChargeParameterDiscovery_009, TC_EVCC_AC_VTB_ChargeParameterDiscovery_010, TC_EVCC_DC_VTB_ChargeParameterDiscovery_002, TC_EVCC_DC_VTB_ChargeParameterDiscovery_003, TC_EVCC_DC_VTB_ChargeParameterDiscovery_004, TC_EVCC_DC_VTB_ChargeParameterDiscovery_005, TC_EVCC_DC_VTB_ChargeParameterDiscovery_006, TC_EVCC_DC_VTB_ChargeParameterDiscovery_007, TC_EVCC_DC_VTB_ChargeParameterDiscovery_009, TC_EVCC_DC_VTB_ChargeParameterDiscovery_010, TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001, TC_EVCC_CMN_VTB_ChargeParameterDiscovery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_003, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_004, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_005, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_006, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_010, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_011, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_012, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_003, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_004, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_005, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_006, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_010, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_011, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_012, TC_EVCC_AC_VTB_ChargingStatus_002, TC_EVCC_AC_VTB_ChargingStatus_003, TC_EVCC_AC_VTB_ChargingStatus_004, TC_EVCC_AC_VTB_ChargingStatus_005, TC_EVCC_AC_VTB_ChargingStatus_006, TC_EVCC_AC_VTB_ChargingStatus_007, TC_EVCC_AC_VTB_ChargingStatus_008, TC_EVCC_DC_VTB_ChargingStatus_002, TC_EVCC_DC_VTB_ChargingStatus_003, TC_EVCC_DC_VTB_ChargingStatus_004, TC_EVCC_DC_VTB_ChargingStatus_005, TC_EVCC_DC_VTB_ChargingStatus_006, TC_EVCC_DC_VTB_ChargingStatus_007, TC_EVCC_DC_VTB_ChargingStatus_008, TC_EVCC_AC_VTB_MeteringReceipt_002, TC_EVCC_AC_VTB_MeteringReceipt_003, TC_EVCC_AC_VTB_MeteringReceipt_004, TC_EVCC_AC_VTB_MeteringReceipt_005, TC_EVCC_AC_VTB_MeteringReceipt_006, TC_EVCC_AC_VTB_MeteringReceipt_007, TC_EVCC_AC_VTB_MeteringReceipt_010, TC_EVCC_AC_VTB_MeteringReceipt_011, TC_EVCC_AC_VTB_MeteringReceipt_012, TC_EVCC_DC_VTB_MeteringReceipt_002,

Group name	TC IDs
	TC_EVCC_DC_VTB_MeteringReceipt_003, TC_EVCC_DC_VTB_MeteringReceipt_004, TC_EVCC_DC_VTB_MeteringReceipt_005, TC_EVCC_DC_VTB_MeteringReceipt_006, TC_EVCC_DC_VTB_MeteringReceipt_007, TC_EVCC_DC_VTB_MeteringReceipt_010, TC_EVCC_DC_VTB_MeteringReceipt_011, TC_EVCC_DC_VTB_MeteringReceipt_012, TC_EVCC_AC_VTB_PowerDelivery_002, TC_EVCC_AC_VTB_PowerDelivery_003, TC_EVCC_AC_VTB_PowerDelivery_004, TC_EVCC_AC_VTB_PowerDelivery_005, TC_EVCC_AC_VTB_PowerDelivery_006, TC_EVCC_AC_VTB_PowerDelivery_007, TC_EVCC_AC_VTB_PowerDelivery_008, TC_EVCC_AC_VTB_PowerDelivery_009, TC_EVCC_AC_VTB_PowerDelivery_010, TC_EVCC_AC_VTB_PowerDelivery_011, TC_EVCC_AC_VTB_PowerDelivery_012, TC_EVCC_DC_VTB_PreCharge_002, TC_EVCC_DC_VTB_PreCharge_003, TC_EVCC_DC_VTB_PreCharge_004, TC_EVCC_DC_VTB_PreCharge_005, TC_EVCC_DC_VTB_PreCharge_006, TC_EVCC_DC_VTB_PreCharge_007, TC_EVCC_DC_VTB_PreCharge_008, TC_EVCC_DC_VTB_PreCharge_009, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_002, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_003, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_004, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_005, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_006, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_007, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_008, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_010, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_011, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_012, TC_EVCC_CMN_VTB_ServiceDetail_003, TC_EVCC_CMN_VTB_ServiceDetail_004, TC_EVCC_CMN_VTB_ServiceDetail_005, TC_EVCC_CMN_VTB_ServiceDetail_006, TC_EVCC_CMN_VTB_ServiceDetail_007, TC_EVCC_CMN_VTB_ServiceDetail_008, TC_EVCC_CMN_VTB_ServiceDetail_009, TC_EVCC_CMN_VTB_ServiceDetail_010, TC_EVCC_CMN_VTB_ServiceDetail_011, TC_EVCC_CMN_VTB_ServiceDetail_012, TC_EVCC_CMN_VTB_ServiceDetail_013, TC_EVCC_CMN_VTB_ServiceDetail_014, TC_EVCC_CMN_VTB_ServiceDetail_015, TC_EVCC_CMN_VTB_ServiceDetail_016, TC_EVCC_CMN_VTB_ServiceDetail_017, TC_EVCC_CMN_VTB_ServiceDetail_018, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_010, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_011, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_012, TC_EVCC_CMN_VTB_ServiceDiscovery_002, TC_EVCC_CMN_VTB_ServiceDiscovery_003, TC_EVCC_CMN_VTB_ServiceDiscovery_004, TC_EVCC_CMN_VTB_ServiceDiscovery_005, TC_EVCC_CMN_VTB_ServiceDiscovery_006, TC_EVCC_CMN_VTB_ServiceDiscovery_007, TC_EVCC_CMN_VTB_ServiceDiscovery_008, TC_EVCC_CMN_VTB_SessionSetup_002, TC_EVCC_CMN_VTB_SessionSetup_003, TC_EVCC_CMN_VTB_SessionSetup_004, TC_EVCC_CMN_VTB_SessionSetup_005, TC_EVCC_CMN_VTB_SessionSetup_006, TC_EVCC_CMN_VTB_SessionSetup_007, TC_EVCC_AC_VTB_SessionStop_002, TC_EVCC_AC_VTB_SessionStop_003, TC_EVCC_AC_VTB_SessionStop_004, TC_EVCC_AC_VTB_SessionStop_005, TC_EVCC_AC_VTB_SessionStop_006, TC_EVCC_CMN_VTB_SupportedAppProtocol_002, TC_EVCC_CMN_VTB_SupportedAppProtocol_003, TC_EVCC_DC_VTB_WeldingDetection_002, TC_EVCC_DC_VTB_WeldingDetection_003, TC_EVCC_DC_VTB_WeldingDetection_004, TC_EVCC_DC_VTB_WeldingDetection_005, TC_EVCC_DC_VTB_WeldingDetection_006, TC_EVCC_DC_VTB_WeldingDetection_007, TC_EVCC_DC_VTB_WeldingDetection_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_002, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_003, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_004, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_006,
Group_[V2G2-214]_[V2G2-216]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012,

	TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026,
-278]_[V2G2-279]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026,

Group name	TC IDs
	TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032,

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_WeldingDetection_001,
Group_[V2G2-360]_[V2G2-361]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026,
Group_[V2G2-370]_[V2G2-371]	TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011,

Group name	TC IDs
	TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026,
Group_[V2G2-848]_[V2G2-849]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014,
Group_[V2G2-840]_[V2G2-841]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014,
Group_[V2G2-374]_[V2G2-375]	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022,
Group_[V2G2-257]_[V2G2-258]	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_DC_VTB_CurrentDemand_001,
Group_[V2G2-005]_[V2G2-007]	TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_CMN_VTB_CertificateInstallation_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_001,

Group name	TC IDs
	TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_DC_VTB_MeteringReceipt_001,
Group_[V2G2-245]_[V2G2-246]	TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021,
Group_[V2G2-276]_[V2G2-277]	TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_009,
Group_[V2G2-512]_[V2G2-408]	TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021,
Group_[V2G2-790]_[V2G2-789]	TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021,
Group_[V2G2-293]_[V2G2-673]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027,

Group name	TC IDs
	TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022,
Group_[V2G2-297]_[V2G2-298]	TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020,
Group_[V2G2-253]_[V2G2-254]	TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022,
Group_[V2G2-239]_[V2G2-738]	TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_008, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_006, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015,

Group name	TC IDs
	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020,
Group_[V2G2-718]_[V2G2-739]	TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020,
Group_[V2G2-913]_[V2G2-914]	TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_006, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020,
Group_[V2G2-082]_[V2G2-086]	TC_EVCC_CMN_VTB_V2GTPSDP_001, TC_EVCC_CMN_VTB_V2GTPSDP_002, TC_EVCC_CMN_VTB_V2GTPSDP_003, TC_EVCC_CMN_VTB_V2GTPSDP_004, TC_EVCC_CMN_VTB_V2GTPSDP_005, TC_EVCC_CMN_VTB_V2GTPSessionSetup_001, TC_EVCC_CMN_VTB_V2GTPSessionSetup_002, TC_EVCC_CMN_VTB_V2GTPSessionSetup_003,

Group name	TC IDs
	TC_EVCC_CMN_VTB_V2GTPSessionSetup_004, TC_EVCC_CMN_VTB_V2GTPSessionSetup_005, TC_SECC_CMN_VTB_V2GTPSDP_001, TC_SECC_CMN_VTB_V2GTPSDP_002, TC_SECC_CMN_VTB_V2GTPSDP_003, TC_SECC_CMN_VTB_V2GTPSDP_004, TC_SECC_CMN_VTB_V2GTPSDP_005, TC_SECC_CMN_VTB_V2GTPSessionSetup_001, TC_SECC_CMN_VTB_V2GTPSessionSetup_002, TC_SECC_CMN_VTB_V2GTPSessionSetup_003, TC_SECC_CMN_VTB_V2GTPSessionSetup_004, TC_SECC_CMN_VTB_V2GTPSessionSetup_005,
Group_[V2G2-729]_[V2G2-034]	TC_SECC_CMN_VTB_Authorization_003, TC_SECC_CMN_VTB_Authorization_004, TC_SECC_CMN_VTB_Authorization_005, TC_SECC_CMN_VTB_Authorization_006, TC_SECC_CMN_VTB_Authorization_007, TC_SECC_CMN_VTB_Authorization_008, TC_SECC_CMN_VTB_Authorization_009, TC_SECC_CMN_VTB_Authorization_010, TC_SECC_CMN_VTB_Authorization_011, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_DC_VTB_CableCheck_003, TC_SECC_DC_VTB_CableCheck_004, TC_SECC_DC_VTB_CableCheck_005, TC_SECC_DC_VTB_CableCheck_008, TC_SECC_CMN_VTB_CertificateInstallation_002, TC_SECC_CMN_VTB_CertificateInstallation_003, TC_SECC_CMN_VTB_CertificateInstallation_004, TC_SECC_CMN_VTB_CertificateInstallation_005, TC_SECC_CMN_VTB_CertificateInstallation_006, TC_SECC_CMN_VTB_CertificateInstallation_007, TC_SECC_CMN_VTB_CertificateInstallation_008, TC_SECC_CMN_VTB_CertificateUpdate_002, TC_SECC_CMN_VTB_CertificateUpdate_003, TC_SECC_CMN_VTB_CertificateUpdate_004, TC_SECC_CMN_VTB_CertificateUpdate_005, TC_SECC_CMN_VTB_CertificateUpdate_006, TC_SECC_CMN_VTB_CertificateUpdate_007, TC_SECC_CMN_VTB_CertificateUpdate_008, TC_SECC_CMN_VTB_CertificateUpdate_009, TC_SECC_CMN_VTB_CertificateUpdate_010, TC_SECC_CMN_VTB_CertificateUpdate_011, TC_SECC_CMN_VTB_CertificateUpdate_012, TC_SECC_CMN_VTB_CertificateUpdate_013, TC_SECC_CMN_VTB_CertificateUpdate_014, TC_SECC_CMN_VTB_CertificateUpdate_015, TC_SECC_CMN_VTB_CertificateUpdate_016, TC_SECC_CMN_VTB_CertificateUpdate_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_003, TC_SECC_AC_VTB_ChargeParameterDiscovery_004, TC_SECC_AC_VTB_ChargeParameterDiscovery_005, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_002, TC_SECC_DC_VTB_ChargeParameterDiscovery_003, TC_SECC_DC_VTB_ChargeParameterDiscovery_004, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_CMN_VTB_ChargeParameterDiscovery_001, TC_SECC_CMN_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargingStatus_003, TC_SECC_AC_VTB_ChargingStatus_004, TC_SECC_AC_VTB_ChargingStatus_005, TC_SECC_DC_VTB_CurrentDemand_003, TC_SECC_DC_VTB_CurrentDemand_004, TC_SECC_DC_VTB_CurrentDemand_005, TC_SECC_AC_VTB_MeteringReceipt_002, TC_SECC_AC_VTB_MeteringReceipt_003, TC_SECC_AC_VTB_MeteringReceipt_004, TC_SECC_AC_VTB_MeteringReceipt_005, TC_SECC_AC_VTB_MeteringReceipt_006, TC_SECC_AC_VTB_MeteringReceipt_007, TC_SECC_DC_VTB_MeteringReceipt_002, TC_SECC_DC_VTB_MeteringReceipt_003, TC_SECC_DC_VTB_MeteringReceipt_004, TC_SECC_DC_VTB_MeteringReceipt_005, TC_SECC_DC_VTB_MeteringReceipt_006, TC_SECC_DC_VTB_MeteringReceipt_007, TC_SECC_CMN_VTB_PaymentDetails_004, TC_SECC_CMN_VTB_PaymentDetails_005, TC_SECC_CMN_VTB_PaymentDetails_006, TC_SECC_CMN_VTB_PaymentDetails_007, TC_SECC_CMN_VTB_PaymentDetails_008, TC_SECC_CMN_VTB_PaymentDetails_009, TC_SECC_CMN_VTB_PaymentDetails_010, TC_SECC_CMN_VTB_PaymentServiceSelection_004, TC_SECC_CMN_VTB_PaymentServiceSelection_005, TC_SECC_CMN_VTB_PaymentServiceSelection_006, TC_SECC_CMN_VTB_PaymentServiceSelection_007, TC_SECC_CMN_VTB_PaymentServiceSelection_008, TC_SECC_CMN_VTB_PaymentServiceSelection_009, TC_SECC_CMN_VTB_PaymentServiceSelection_010, TC_SECC_AC_VTB_PowerDelivery_006, TC_SECC_AC_VTB_PowerDelivery_007, TC_SECC_AC_VTB_PowerDelivery_008, TC_SECC_AC_VTB_PowerDelivery_009, TC_SECC_AC_VTB_PowerDelivery_010, TC_SECC_AC_VTB_PowerDelivery_011, TC_SECC_AC_VTB_PowerDelivery_012, TC_SECC_AC_VTB_PowerDelivery_013,

Group name	TC IDs
	TC_SECC_AC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_003, TC_SECC_DC_VTB_PowerDelivery_004, TC_SECC_DC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_006, TC_SECC_DC_VTB_PowerDelivery_007, TC_SECC_DC_VTB_PowerDelivery_008, TC_SECC_DC_VTB_PowerDelivery_009, TC_SECC_DC_VTB_PowerDelivery_010, TC_SECC_DC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_PreCharge_002, TC_SECC_DC_VTB_PreCharge_003, TC_SECC_DC_VTB_PreCharge_004, TC_SECC_CMN_VTB_ServiceDetail_003, TC_SECC_CMN_VTB_ServiceDetail_004, TC_SECC_CMN_VTB_ServiceDetail_005, TC_SECC_CMN_VTB_ServiceDetail_006, TC_SECC_CMN_VTB_ServiceDetail_007, TC_SECC_CMN_VTB_ServiceDetail_008, TC_SECC_CMN_VTB_ServiceDetail_009, TC_SECC_CMN_VTB_ServiceDetail_010, TC_SECC_CMN_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_ServiceDiscovery_003, TC_SECC_CMN_VTB_ServiceDiscovery_004, TC_SECC_CMN_VTB_SessionSetup_002, TC_SECC_CMN_VTB_SessionSetup_003, TC_SECC_CMN_VTB_SessionSetup_004, TC_SECC_AC_VTB_SessionStop_002, TC_SECC_AC_VTB_SessionStop_003, TC_SECC_AC_VTB_SessionStop_004, TC_SECC_DC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_004, TC_SECC_DC_VTB_SessionStop_005, TC_SECC_DC_VTB_SessionStop_006, TC_SECC_DC_VTB_SessionStop_007, TC_SECC_DC_VTB_SessionStop_008, TC_SECC_DC_VTB_SessionStop_009, TC_SECC_CMN_VTB_SupportedAppProtocol_002, TC_SECC_CMN_VTB_SupportedAppProtocol_003, TC_SECC_DC_VTB_WeldingDetection_002, TC_SECC_DC_VTB_WeldingDetection_003, TC_SECC_DC_VTB_WeldingDetection_004, TC_SECC_DC_VTB_WeldingDetection_005,
Group_[V2G2-212]_[V2G2-213]	TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_005, TC_SECC_CMN_VTB_Authorization_006, TC_SECC_CMN_VTB_Authorization_007, TC_SECC_CMN_VTB_Authorization_010, TC_SECC_CMN_VTB_Authorization_011, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_CMN_VTB_ServiceDiscovery_002,
Group_[V2G2-539]_[V2G2-736]	TC_SECC_CMN_VTB_Authorization_003, TC_SECC_CMN_VTB_Authorization_004, TC_SECC_CMN_VTB_Authorization_005, TC_SECC_CMN_VTB_Authorization_006, TC_SECC_CMN_VTB_Authorization_007, TC_SECC_CMN_VTB_Authorization_010, TC_SECC_CMN_VTB_Authorization_011, TC_SECC_DC_VTB_CableCheck_003, TC_SECC_DC_VTB_CableCheck_004, TC_SECC_DC_VTB_CableCheck_008, TC_SECC_CMN_VTB_CertificateInstallation_002, TC_SECC_CMN_VTB_CertificateInstallation_003, TC_SECC_CMN_VTB_CertificateInstallation_005, TC_SECC_CMN_VTB_CertificateInstallation_006, TC_SECC_CMN_VTB_CertificateInstallation_007, TC_SECC_CMN_VTB_CertificateInstallation_008, TC_SECC_CMN_VTB_CertificateUpdate_002, TC_SECC_CMN_VTB_CertificateUpdate_003, TC_SECC_CMN_VTB_CertificateUpdate_005, TC_SECC_CMN_VTB_CertificateUpdate_006, TC_SECC_CMN_VTB_CertificateUpdate_007, TC_SECC_CMN_VTB_CertificateUpdate_008, TC_SECC_CMN_VTB_CertificateUpdate_009, TC_SECC_CMN_VTB_CertificateUpdate_010, TC_SECC_CMN_VTB_CertificateUpdate_011, TC_SECC_CMN_VTB_CertificateUpdate_012, TC_SECC_CMN_VTB_CertificateUpdate_013, TC_SECC_CMN_VTB_CertificateUpdate_014, TC_SECC_CMN_VTB_CertificateUpdate_015, TC_SECC_CMN_VTB_CertificateUpdate_016, TC_SECC_CMN_VTB_CertificateUpdate_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_003, TC_SECC_AC_VTB_ChargeParameterDiscovery_004, TC_SECC_AC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_002, TC_SECC_DC_VTB_ChargeParameterDiscovery_003, TC_SECC_DC_VTB_ChargeParameterDiscovery_004, TC_SECC_CMN_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargingStatus_003, TC_SECC_AC_VTB_ChargingStatus_004, TC_SECC_DC_VTB_CurrentDemand_003, TC_SECC_DC_VTB_CurrentDemand_004, TC_SECC_AC_VTB_MeteringReceipt_002, TC_SECC_AC_VTB_MeteringReceipt_003, TC_SECC_AC_VTB_MeteringReceipt_004, TC_SECC_AC_VTB_MeteringReceipt_006, TC_SECC_AC_VTB_MeteringReceipt_007, TC_SECC_DC_VTB_MeteringReceipt_002, TC_SECC_DC_VTB_MeteringReceipt_003, TC_SECC_DC_VTB_MeteringReceipt_004, TC_SECC_DC_VTB_MeteringReceipt_006, TC_SECC_DC_VTB_MeteringReceipt_007, TC_SECC_CMN_VTB_PaymentDetails_004, TC_SECC_CMN_VTB_PaymentDetails_005, TC_SECC_CMN_VTB_PaymentDetails_007, TC_SECC_CMN_VTB_PaymentDetails_008, TC_SECC_CMN_VTB_PaymentDetails_009, TC_SECC_CMN_VTB_PaymentDetails_010,

Group name	TC IDs
	TC_SECC_CMN_VTB_PaymentServiceSelection_004, TC_SECC_CMN_VTB_PaymentServiceSelection_005, TC_SECC_CMN_VTB_PaymentServiceSelection_006, TC_SECC_CMN_VTB_PaymentServiceSelection_007, TC_SECC_CMN_VTB_PaymentServiceSelection_009, TC_SECC_CMN_VTB_PaymentServiceSelection_010, TC_SECC_AC_VTB_PowerDelivery_006, TC_SECC_AC_VTB_PowerDelivery_007, TC_SECC_AC_VTB_PowerDelivery_008, TC_SECC_AC_VTB_PowerDelivery_009, TC_SECC_AC_VTB_PowerDelivery_010, TC_SECC_AC_VTB_PowerDelivery_011, TC_SECC_AC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_003, TC_SECC_DC_VTB_PowerDelivery_004, TC_SECC_DC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_006, TC_SECC_DC_VTB_PowerDelivery_007, TC_SECC_DC_VTB_PowerDelivery_008, TC_SECC_DC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_PreCharge_002, TC_SECC_DC_VTB_PreCharge_003, TC_SECC_CMN_VTB_ServiceDetail_003, TC_SECC_CMN_VTB_ServiceDetail_004, TC_SECC_CMN_VTB_ServiceDetail_005, TC_SECC_CMN_VTB_ServiceDetail_006, TC_SECC_CMN_VTB_ServiceDetail_007, TC_SECC_CMN_VTB_ServiceDetail_008, TC_SECC_CMN_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_ServiceDiscovery_003, TC_SECC_CMN_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionStop_002, TC_SECC_AC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_004, TC_SECC_DC_VTB_SessionStop_006, TC_SECC_DC_VTB_SessionStop_007, TC_SECC_DC_VTB_SessionStop_008, TC_SECC_CMN_VTB_SupportedAppProtocol_002, TC_SECC_DC_VTB_WeldingDetection_002, TC_SECC_DC_VTB_WeldingDetection_003, TC_SECC_DC_VTB_WeldingDetection_005,
Group_[V2G2-563]_[V2G2-856]	TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017,
Group_[V2G2-251]_[V2G2-252]	TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_004, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_CableCheck_008,
Group_[V2G2-103]_[V2G2-104]	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateUpdate_001,
Group_[V2G2-237]_[V2G2-238]	TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateInstallation_003, TC_SECC_CMN_VTB_CertificateInstallation_005, TC_SECC_CMN_VTB_CertificateInstallation_006, TC_SECC_CMN_VTB_CertificateInstallation_007, TC_SECC_CMN_VTB_CertificateInstallation_008,
Group_[V2G2-231]_[V2G2-232]	TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_CMN_VTB_CertificateUpdate_003, TC_SECC_CMN_VTB_CertificateUpdate_005, TC_SECC_CMN_VTB_CertificateUpdate_006, TC_SECC_CMN_VTB_CertificateUpdate_007, TC_SECC_CMN_VTB_CertificateUpdate_008, TC_SECC_CMN_VTB_CertificateUpdate_009, TC_SECC_CMN_VTB_CertificateUpdate_010, TC_SECC_CMN_VTB_CertificateUpdate_011, TC_SECC_CMN_VTB_CertificateUpdate_012, TC_SECC_CMN_VTB_CertificateUpdate_013, TC_SECC_CMN_VTB_CertificateUpdate_014, TC_SECC_CMN_VTB_CertificateUpdate_015, TC_SECC_CMN_VTB_CertificateUpdate_016, TC_SECC_CMN_VTB_CertificateUpdate_017,
Group_[V2G2-218]_[V2G2-220]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_003, TC_SECC_AC_VTB_ChargeParameterDiscovery_004, TC_SECC_AC_VTB_ChargeParameterDiscovery_005, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012,

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_002, TC_SECC_DC_VTB_ChargeParameterDiscovery_003, TC_SECC_DC_VTB_ChargeParameterDiscovery_004, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-358]_[V2G2-359]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009,

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005,
Group_[V2G2-931]_[V2G2-708]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016,
Group_[V2G2-741]_[V2G2-745]	TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021,

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_AC_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_004, TC_SECC_DC_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_004,
Group_[V2G2-372]_[V2G2-373]	TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-712]_[V2G2-713]	TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_014,
Group_[V2G2-195]_[V2G2-196]	TC_SECC_CMN_VTB_Authorization_003, TC_SECC_CMN_VTB_Authorization_004, TC_SECC_DC_VTB_CableCheck_003, TC_SECC_CMN_VTB_CertificateInstallation_002, TC_SECC_CMN_VTB_CertificateUpdate_002, TC_SECC_CMN_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargingStatus_003, TC_SECC_DC_VTB_CurrentDemand_003, TC_SECC_AC_VTB_MeteringReceipt_002, TC_SECC_DC_VTB_MeteringReceipt_002, TC_SECC_CMN_VTB_PaymentDetails_004, TC_SECC_CMN_VTB_PaymentServiceSelection_004, TC_SECC_AC_VTB_PowerDelivery_006, TC_SECC_AC_VTB_PowerDelivery_007, TC_SECC_DC_VTB_PowerDelivery_003, TC_SECC_DC_VTB_PowerDelivery_004, TC_SECC_DC_VTB_PreCharge_002, TC_SECC_CMN_VTB_ServiceDetail_003, TC_SECC_CMN_VTB_ServiceDetail_004, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_ServiceDiscovery_003, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionStop_002, TC_SECC_DC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_007, TC_SECC_DC_VTB_WeldingDetection_002,
Group_[V2G2-459]_[V2G2-538]	TC_SECC_CMN_VTB_Authorization_003, TC_SECC_CMN_VTB_Authorization_004, TC_SECC_DC_VTB_CableCheck_003, TC_SECC_CMN_VTB_CertificateInstallation_002, TC_SECC_CMN_VTB_CertificateUpdate_002, TC_SECC_CMN_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargingStatus_003, TC_SECC_DC_VTB_CurrentDemand_003, TC_SECC_AC_VTB_MeteringReceipt_002, TC_SECC_DC_VTB_MeteringReceipt_002, TC_SECC_CMN_VTB_PaymentDetails_004,

Group name	TC IDs
	TC_SECC_CMN_VTB_PaymentServiceSelection_004, TC_SECC_AC_VTB_PowerDelivery_006, TC_SECC_AC_VTB_PowerDelivery_007, TC_SECC_DC_VTB_PowerDelivery_003, TC_SECC_DC_VTB_PowerDelivery_004, TC_SECC_DC_VTB_PreCharge_002, TC_SECC_CMN_VTB_ServiceDetail_003, TC_SECC_CMN_VTB_ServiceDetail_004, TC_SECC_CMN_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionStop_002, TC_SECC_DC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_007, TC_SECC_DC_VTB_WeldingDetection_002,
Group_[V2G2-243]_[V2G2-244]	TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_004, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_009,
Group_[V2G2-575]_[V2G2-577]	TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_009,
Group_[V2G2-259]_[V2G2-260]	TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_004, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_009,
Group_[V2G2-247]_[V2G2-248]	TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_AC_VTB_MeteringReceipt_003, TC_SECC_AC_VTB_MeteringReceipt_004, TC_SECC_AC_VTB_MeteringReceipt_006, TC_SECC_AC_VTB_MeteringReceipt_007, TC_SECC_DC_VTB_MeteringReceipt_001, TC_SECC_DC_VTB_MeteringReceipt_003, TC_SECC_DC_VTB_MeteringReceipt_004, TC_SECC_DC_VTB_MeteringReceipt_006, TC_SECC_DC_VTB_MeteringReceipt_007,
Group_[V2G2-208]_[V2G2-209]	TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentDetails_005, TC_SECC_CMN_VTB_PaymentDetails_007, TC_SECC_CMN_VTB_PaymentDetails_008, TC_SECC_CMN_VTB_PaymentDetails_009, TC_SECC_CMN_VTB_PaymentDetails_010,
Group_[V2G2-203]_[V2G2-204]	TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_CMN_VTB_PaymentServiceSelection_005, TC_SECC_CMN_VTB_PaymentServiceSelection_006, TC_SECC_CMN_VTB_PaymentServiceSelection_007, TC_SECC_CMN_VTB_PaymentServiceSelection_009, TC_SECC_CMN_VTB_PaymentServiceSelection_010,
Group_[V2G2-223]_[V2G2-226]	TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_PowerDelivery_008, TC_SECC_AC_VTB_PowerDelivery_009, TC_SECC_AC_VTB_PowerDelivery_010, TC_SECC_AC_VTB_PowerDelivery_011, TC_SECC_AC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_006, TC_SECC_DC_VTB_PowerDelivery_007, TC_SECC_DC_VTB_PowerDelivery_008, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021,
Group_[V2G2-199]_[V2G2-200]	TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_005, TC_SECC_CMN_VTB_ServiceDetail_006, TC_SECC_CMN_VTB_ServiceDetail_007, TC_SECC_CMN_VTB_ServiceDetail_008, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_ServiceDetail_012,
Group_[V2G2-240]_[V2G2-241]	TC_SECC_AC_VTB_SessionStop_001, TC_SECC_AC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_DC_VTB_SessionStop_004, TC_SECC_DC_VTB_SessionStop_006, TC_SECC_DC_VTB_SessionStop_008,
Group_[V2G2-165]_[V2G2-178]	TC_EVCC_CMN_VTB_SupportedAppProtocol_001, TC_EVCC_CMN_VTB_SupportedAppProtocol_004,

Group name	TC IDs
	TC_EVCC_CMN_VTB_SupportedAppProtocol_005, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_002, TC_SECC_CMN_VTB_SupportedAppProtocol_004, TC_SECC_CMN_VTB_SupportedAppProtocol_006,
Group_[V2G2-157]_[V2G2-160]_[V2G2-161]	TC_EVCC_CMN_VTB_SDP_005, TC_EVCC_AC_VTB_SDP_001, TC_EVCC_AC_VTB_SDP_002, TC_EVCC_AC_VTB_SDP_003, TC_EVCC_AC_VTB_SDP_004, TC_EVCC_AC_VTB_SDP_005, TC_EVCC_AC_VTB_SDP_006, TC_EVCC_AC_VTB_SDP_007, TC_EVCC_AC_VTB_SDP_008, TC_EVCC_AC_VTB_SDP_009, TC_EVCC_AC_VTB_SDP_010, TC_EVCC_AC_VTB_SDP_011, TC_EVCC_AC_VTB_SDP_012, TC_EVCC_AC_VTB_SDP_013, TC_EVCC_AC_VTB_SDP_014, TC_EVCC_AC_VTB_SDP_015, TC_EVCC_AC_VTB_SDP_016, TC_EVCC_DC_VTB_SDP_001, TC_EVCC_DC_VTB_SDP_002, TC_EVCC_DC_VTB_SDP_003, TC_EVCC_DC_VTB_SDP_004, TC_EVCC_DC_VTB_SDP_005, TC_EVCC_DC_VTB_SDP_006, TC_EVCC_DC_VTB_SDP_007, TC_EVCC_DC_VTB_SDP_008, TC_EVCC_DC_VTB_SDP_009, TC_EVCC_DC_VTB_SDP_010, TC_EVCC_DC_VTB_SDP_011, TC_EVCC_DC_VTB_SDP_012, TC_EVCC_DC_VTB_SDP_013, TC_EVCC_DC_VTB_SDP_014, TC_EVCC_DC_VTB_SDP_015, TC_EVCC_DC_VTB_SDP_016,
Group_[V2G2-367]_[V2G2-368]_[V2G2-369]	TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009,
Group_[V2G2-221]_[V2G2-222]_[V2G2-286]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009,

Group name	TC IDs
	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022,
Group_[V2G2-006]_[V2G2-652]_[V2G2-909]	TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_CMN_VTB_CertificateInstallation_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002,
Group_[V2G2-493]_[V2G2-494]_[V2G2-432]	TC_EVCC_CMN_VTB_PaymentServiceSelection_001, TC_EVCC_CMN_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_PaymentServiceSelection_015, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_013, TC_EVCC_DC_VTB_PaymentServiceSelection_014,
Group_[V2G2-510]_[V2G2-847]_[V2G2-846]	TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021,
Group_[V2G2-188]_[V2G2-189]_[V2G2-485]	TC_EVCC_CMN_VTB_SessionSetup_001, TC_EVCC_CMN_VTB_SessionSetup_008, TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002,

Group name	TC IDs
	TC_EVCC_AC_VTB_SessionSetup_003, TC_EVCC_AC_VTB_SessionSetup_004, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002, TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004,
Group_[V2G2-633]_[V2G2-635]_[V2G2-637]	TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019,
Group_[V2G2-664]_[V2G2-666]_[V2G2-667]	TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011,

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_DC_VTB_MeteringReceipt_001, TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_ServiceDetail_012, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_004, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_DC_VTB_WeldingDetection_001,
Group_[V2G2-573]_[V2G2-362]_[V2G2-363]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013,

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-364]_[V2G2-365]_[V2G2-366]	TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019,

Group name	TC IDs
	TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_WeldingDetection_001,
Group_[V2G2-441]_[V2G2-442]_[V2G2-443]	TC_SECC_CMN_VTB_Authorization_008, TC_SECC_CMN_VTB_Authorization_009, TC_SECC_DC_VTB_CableCheck_005, TC_SECC_CMN_VTB_CertificateInstallation_004, TC_SECC_CMN_VTB_CertificateUpdate_004, TC_SECC_CMN_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargingStatus_005, TC_SECC_DC_VTB_CurrentDemand_005, TC_SECC_AC_VTB_MeteringReceipt_005, TC_SECC_DC_VTB_MeteringReceipt_005, TC_SECC_CMN_VTB_PaymentDetails_006, TC_SECC_CMN_VTB_PaymentServiceSelection_008, TC_SECC_AC_VTB_PowerDelivery_012, TC_SECC_AC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_PowerDelivery_009, TC_SECC_DC_VTB_PowerDelivery_010, TC_SECC_DC_VTB_PreCharge_004, TC_SECC_CMN_VTB_ServiceDetail_009, TC_SECC_CMN_VTB_ServiceDetail_010, TC_SECC_CMN_VTB_ServiceDiscovery_004, TC_SECC_CMN_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionStop_004, TC_SECC_DC_VTB_SessionStop_005, TC_SECC_DC_VTB_SessionStop_009, TC_SECC_CMN_VTB_SupportedAppProtocol_003, TC_SECC_DC_VTB_WeldingDetection_004,
Group_[V2G2-142]_[V2G2-623]_[V2G2-018]_[V2G2-019]	TC_EVCC_CMN_VTB_SDP_001, TC_EVCC_CMN_VTB_SDP_002, TC_EVCC_CMN_VTB_SDP_003, TC_EVCC_CMN_VTB_SDP_004, TC_EVCC_CMN_VTB_SDP_005, TC_EVCC_CMN_VTB_SDP_006, TC_EVCC_AC_VTB_SDP_001, TC_EVCC_AC_VTB_SDP_002, TC_EVCC_AC_VTB_SDP_003, TC_EVCC_AC_VTB_SDP_004, TC_EVCC_AC_VTB_SDP_005, TC_EVCC_AC_VTB_SDP_006, TC_EVCC_AC_VTB_SDP_007, TC_EVCC_AC_VTB_SDP_008, TC_EVCC_AC_VTB_SDP_009, TC_EVCC_AC_VTB_SDP_010, TC_EVCC_AC_VTB_SDP_011, TC_EVCC_AC_VTB_SDP_012, TC_EVCC_AC_VTB_SDP_013, TC_EVCC_AC_VTB_SDP_014, TC_EVCC_AC_VTB_SDP_015, TC_EVCC_AC_VTB_SDP_016, TC_EVCC_DC_VTB_SDP_001, TC_EVCC_DC_VTB_SDP_002, TC_EVCC_DC_VTB_SDP_003, TC_EVCC_DC_VTB_SDP_004, TC_EVCC_DC_VTB_SDP_005, TC_EVCC_DC_VTB_SDP_006, TC_EVCC_DC_VTB_SDP_007, TC_EVCC_DC_VTB_SDP_008, TC_EVCC_DC_VTB_SDP_009, TC_EVCC_DC_VTB_SDP_010, TC_EVCC_DC_VTB_SDP_011, TC_EVCC_DC_VTB_SDP_012, TC_EVCC_DC_VTB_SDP_013, TC_EVCC_DC_VTB_SDP_014, TC_EVCC_DC_VTB_SDP_015, TC_EVCC_DC_VTB_SDP_016,
Group_[V2G2-179]_[V2G2-180]_[V2G2-181]_[V2G2-183]	TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_002, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001, TC_EVCC_CMN_VTB_CertificateInstallation_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016,

Group name	TC IDs
	TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_AC_VTB_ChargingStatus_001, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_CMN_VTB_PaymentDetails_003, TC_EVCC_CMN_VTB_PaymentServiceSelection_001, TC_EVCC_CMN_VTB_PaymentServiceSelection_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_003, TC_EVCC_CMN_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_001, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_005, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_009, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_012, TC_EVCC_AC_VTB_PaymentServiceSelection_013, TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_PaymentServiceSelection_015, TC_EVCC_AC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_PaymentServiceSelection_017, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_004, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_007, TC_EVCC_DC_VTB_PaymentServiceSelection_008,

Group name	TC IDs
	TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_011, TC_EVCC_DC_VTB_PaymentServiceSelection_012, TC_EVCC_DC_VTB_PaymentServiceSelection_013, TC_EVCC_DC_VTB_PaymentServiceSelection_014, TC_EVCC_DC_VTB_PaymentServiceSelection_015, TC_EVCC_DC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_CMN_VTB_ServiceDetail_001, TC_EVCC_CMN_VTB_ServiceDetail_002, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_CMN_VTB_ServiceDiscovery_001, TC_EVCC_CMN_VTB_SessionSetup_001, TC_EVCC_CMN_VTB_SessionSetup_008, TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002, TC_EVCC_AC_VTB_SessionSetup_003, TC_EVCC_AC_VTB_SessionSetup_004, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002, TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004, TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_008, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_006, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008,

Group name	TC IDs
	TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020, TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_003, TC_SECC_CMN_VTB_Authorization_004, TC_SECC_CMN_VTB_Authorization_005, TC_SECC_CMN_VTB_Authorization_006, TC_SECC_CMN_VTB_Authorization_007, TC_SECC_CMN_VTB_Authorization_010, TC_SECC_CMN_VTB_Authorization_011, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_003, TC_SECC_DC_VTB_CableCheck_004, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_CableCheck_008, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateInstallation_002, TC_SECC_CMN_VTB_CertificateInstallation_003, TC_SECC_CMN_VTB_CertificateInstallation_005, TC_SECC_CMN_VTB_CertificateInstallation_006, TC_SECC_CMN_VTB_CertificateInstallation_007, TC_SECC_CMN_VTB_CertificateInstallation_008, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_CMN_VTB_CertificateUpdate_002, TC_SECC_CMN_VTB_CertificateUpdate_003, TC_SECC_CMN_VTB_CertificateUpdate_005, TC_SECC_CMN_VTB_CertificateUpdate_006, TC_SECC_CMN_VTB_CertificateUpdate_007, TC_SECC_CMN_VTB_CertificateUpdate_008, TC_SECC_CMN_VTB_CertificateUpdate_009, TC_SECC_CMN_VTB_CertificateUpdate_010, TC_SECC_CMN_VTB_CertificateUpdate_011, TC_SECC_CMN_VTB_CertificateUpdate_012, TC_SECC_CMN_VTB_CertificateUpdate_013, TC_SECC_CMN_VTB_CertificateUpdate_014, TC_SECC_CMN_VTB_CertificateUpdate_015, TC_SECC_CMN_VTB_CertificateUpdate_016, TC_SECC_CMN_VTB_CertificateUpdate_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_003, TC_SECC_AC_VTB_ChargeParameterDiscovery_004, TC_SECC_AC_VTB_ChargeParameterDiscovery_005, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013,

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_002, TC_SECC_DC_VTB_ChargeParameterDiscovery_003, TC_SECC_DC_VTB_ChargeParameterDiscovery_004, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_CMN_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_003, TC_SECC_AC_VTB_ChargingStatus_004, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_003, TC_SECC_DC_VTB_CurrentDemand_004,

Group name	TC IDs
	TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_AC_VTB_MeteringReceipt_002, TC_SECC_AC_VTB_MeteringReceipt_003, TC_SECC_AC_VTB_MeteringReceipt_004, TC_SECC_AC_VTB_MeteringReceipt_006, TC_SECC_AC_VTB_MeteringReceipt_007, TC_SECC_DC_VTB_MeteringReceipt_001, TC_SECC_DC_VTB_MeteringReceipt_002, TC_SECC_DC_VTB_MeteringReceipt_003, TC_SECC_DC_VTB_MeteringReceipt_004, TC_SECC_DC_VTB_MeteringReceipt_006, TC_SECC_DC_VTB_MeteringReceipt_007, TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentDetails_004, TC_SECC_CMN_VTB_PaymentDetails_005, TC_SECC_CMN_VTB_PaymentDetails_007, TC_SECC_CMN_VTB_PaymentDetails_008, TC_SECC_CMN_VTB_PaymentDetails_009, TC_SECC_CMN_VTB_PaymentDetails_010, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_CMN_VTB_PaymentServiceSelection_004, TC_SECC_CMN_VTB_PaymentServiceSelection_005, TC_SECC_CMN_VTB_PaymentServiceSelection_006, TC_SECC_CMN_VTB_PaymentServiceSelection_007, TC_SECC_CMN_VTB_PaymentServiceSelection_009, TC_SECC_CMN_VTB_PaymentServiceSelection_010, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_PowerDelivery_006, TC_SECC_AC_VTB_PowerDelivery_007, TC_SECC_AC_VTB_PowerDelivery_008, TC_SECC_AC_VTB_PowerDelivery_009, TC_SECC_AC_VTB_PowerDelivery_010, TC_SECC_AC_VTB_PowerDelivery_011, TC_SECC_AC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_003, TC_SECC_DC_VTB_PowerDelivery_004, TC_SECC_DC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_006, TC_SECC_DC_VTB_PowerDelivery_007, TC_SECC_DC_VTB_PowerDelivery_008, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_PreCharge_002, TC_SECC_DC_VTB_PreCharge_003, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_003, TC_SECC_CMN_VTB_ServiceDetail_004, TC_SECC_CMN_VTB_ServiceDetail_005, TC_SECC_CMN_VTB_ServiceDetail_006, TC_SECC_CMN_VTB_ServiceDetail_007, TC_SECC_CMN_VTB_ServiceDetail_008, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_ServiceDetail_012, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_ServiceDiscovery_003, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_CMN_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionSetup_004, TC_SECC_AC_VTB_SessionSetup_005, TC_SECC_AC_VTB_SessionSetup_006, TC_SECC_DC_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_002, TC_SECC_DC_VTB_SessionSetup_003, TC_SECC_DC_VTB_SessionSetup_004, TC_SECC_DC_VTB_SessionSetup_005, TC_SECC_DC_VTB_SessionSetup_006, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_AC_VTB_SessionStop_002, TC_SECC_AC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_DC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_004, TC_SECC_DC_VTB_SessionStop_006, TC_SECC_DC_VTB_SessionStop_007, TC_SECC_DC_VTB_SessionStop_008, TC_SECC_DC_VTB_WeldingDetection_001, TC_SECC_DC_VTB_WeldingDetection_002, TC_SECC_DC_VTB_WeldingDetection_003, TC_SECC_DC_VTB_WeldingDetection_005,
Group_[V2G2-659]_[V2G2-660]_[V2G2-661]_[V2G2-662]	TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_002, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001,

Group name	TC IDs
	TC_EVCC_CMN_VTB_CertificateInstallation_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_AC_VTB_ChargingStatus_001, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_CMN_VTB_PaymentDetails_003, TC_EVCC_CMN_VTB_PaymentServiceSelection_001,

Group name	TC IDs
	TC_EVCC_CMN_VTB_PaymentServiceSelection_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_003, TC_EVCC_CMN_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_001, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_005, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_009, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_012, TC_EVCC_AC_VTB_PaymentServiceSelection_013, TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_PaymentServiceSelection_015, TC_EVCC_AC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_PaymentServiceSelection_017, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_004, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_007, TC_EVCC_DC_VTB_PaymentServiceSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_011, TC_EVCC_DC_VTB_PaymentServiceSelection_012, TC_EVCC_DC_VTB_PaymentServiceSelection_013, TC_EVCC_DC_VTB_PaymentServiceSelection_014, TC_EVCC_DC_VTB_PaymentServiceSelection_015, TC_EVCC_DC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_CMN_VTB_ServiceDetail_001, TC_EVCC_CMN_VTB_ServiceDetail_002, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005,

Group name	TC IDs
	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_CMN_VTB_ServiceDiscovery_001, TC_EVCC_CMN_VTB_SessionSetup_001, TC_EVCC_CMN_VTB_SessionSetup_008, TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002, TC_EVCC_AC_VTB_SessionSetup_003, TC_EVCC_AC_VTB_SessionSetup_004, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002, TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004, TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_008, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_006, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030, TC_EVCC_CMN_VTB_SupportedAppProtocol_001, TC_EVCC_CMN_VTB_SupportedAppProtocol_004, TC_EVCC_CMN_VTB_SupportedAppProtocol_005, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020,
Group_[V2G2-434]_[V2G2-436]_[V2G2-437]_[V2G2-438]	TC_EVCC_CMN_VTB_Authorization_008, TC_EVCC_DC_VTB_CableCheck_005, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005, TC_EVCC_CMN_VTB_CertificateInstallation_009, TC_EVCC_CMN_VTB_CertificateUpdate_011, TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007, TC_EVCC_AC_VTB_ChargingStatus_005, TC_EVCC_DC_VTB_CurrentDemand_005,

Group name	TC IDs
	TC_EVCC_AC_VTB_MeteringReceipt_007, TC_EVCC_DC_VTB_MeteringReceipt_007, TC_EVCC_AC_VTB_PowerDelivery_009, TC_EVCC_DC_VTB_PreCharge_005, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009, TC_EVCC_CMN_VTB_ServiceDetail_011, TC_EVCC_CMN_VTB_ServiceDetail_012, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009, TC_EVCC_CMN_VTB_ServiceDiscovery_005, TC_EVCC_CMN_VTB_SessionSetup_004, TC_EVCC_AC_VTB_SessionStop_005, TC_EVCC_CMN_VTB_SupportedAppProtocol_003, TC_EVCC_DC_VTB_WeldingDetection_005, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005,
Group_[V2G2-190]_[V2G2-191]_[V2G2-543]_[V2G2-192]	TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionSetup_004, TC_SECC_AC_VTB_SessionSetup_005, TC_SECC_AC_VTB_SessionSetup_006, TC_SECC_DC_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_002, TC_SECC_DC_VTB_SessionSetup_003, TC_SECC_DC_VTB_SessionSetup_004, TC_SECC_DC_VTB_SessionSetup_005, TC_SECC_DC_VTB_SessionSetup_006,
Group_[V2G2-836]_[V2G2-284]_[V2G2-606]_[V2G2-288]_[V2G2-607]	TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022,
Group_[V2G2-294]_[V2G2-608]_[V2G2-299]_[V2G2-609]_[V2G2-300]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029,

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001, TC_EVCC_CMN_VTB_CertificateInstallation_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateUpdate_001,
Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]	TC_EVCC_CMN_VTB_PaymentServiceSelection_001, TC_EVCC_CMN_VTB_PaymentServiceSelection_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_003, TC_EVCC_CMN_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_001, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_005, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_009, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_012, TC_EVCC_AC_VTB_PaymentServiceSelection_013, TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_PaymentServiceSelection_015,

Group name	TC IDs
	TC_EVCC_AC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_PaymentServiceSelection_017, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_004, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_007, TC_EVCC_DC_VTB_PaymentServiceSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_011, TC_EVCC_DC_VTB_PaymentServiceSelection_012, TC_EVCC_DC_VTB_PaymentServiceSelection_013, TC_EVCC_DC_VTB_PaymentServiceSelection_014, TC_EVCC_DC_VTB_PaymentServiceSelection_015, TC_EVCC_DC_VTB_PaymentServiceSelection_016, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008,
Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_010,

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]	TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_CMN_VTB_CertificateInstallation_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021,
Group_[V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_[V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]	TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_030,

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035,

7.6 Test case description

The test case descriptions in this document are described according to the template shown in Table 31.

Table 31 — Test case description template

TC Id	The TC Id is a unique identifier for a test case. It is specified according to the TC Id naming convention defined in 7.4.2.
Test objective	Short description of test objective according to the requirements from the base standard (15118-2).
Document reference	The document reference indicates the subclauses of the reference standard specifications in which the conformance requirement(s) is/are expressed. The references are provided according to the following format: Document: ISO15118-X:20XX:(IS FDIS) Section(s): x.x.x.x.x, y.y.y.y, ...
Referenced requirement(s)	The referenced requirement(s) refers to the subclauses of the referenced standard specification requirement(s). The requirements are referenced according to the format defined in ISO 15118-2 or ISO 15118-3: [V2GX-XXX], ...

Config Id	The Config Id references the ISO 15118-2 configuration selected for this test case according to 7.3.1. Example: CF_04_001, ...
PICS selection	The PICS selection references the PICS statement(s) for this test case in accordance with 7.3.3. if the corresponding test behavior requires a specific assignment. Example: PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	The PIXIT selection references the PIXIT statement(s) for this test case in accordance with 7.4.6. if the corresponding test behavior requires a specific assignment on SUT side. Example: PIXIT_EVCC_CMN_VAS := serviceDetail
Pre-condition	
The pre-condition defines which state the SUT has to be before executing the actual Test Case. In the corresponding Test Case, when the execution of the initial condition does not succeed, it leads to the assignment of an "Inconclusive" verdict. The pre-condition is defined as reference to the pre-condition function in the ATS. Example: f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
Definition of the events, which are parts of the test case objective, and the SUT are expected to perform in order to conform to the base specification. The expected behavior is defined as reference to the testbehavior function in the ATS. Example: f_SECC_CMN_TB_VTB_ServiceDetail_001	

7.7 Test case specification

The test case specifications are provided in the form of a TTCN-3 test suite according to the TTCN-3 Core Language as defined in ETSI ES 201 873-1 V4.6.1. This subclause defines additional conventions for the scope of this test suite.

7.7.1 Data types

7.7.1.1 V2G PICS

The TTCN-3 data structures defined for PICS (enumerations) according to 7.3.3 are defined in Annex O, O.1.

7.7.1.2 V2G TCP Port Control

The TTCN-3 data structures defined for V2G TCP Port Control according to [V2G4-007] are defined in Annex O, O.3.

7.7.1.3 V2GTP

The TTCN-3 data structures for testing V2GTP conformance are defined in Annex O, O.4. The TTCN-3 data representation for V2GTP message header handling is defined according to ISO 15118-2:2014, 7.8.

7.7.1.4 SDP

The TTCN-3 data structures for testing SDP conformance are defined in Annex O, O.4. The TTCN-3 data representation for SDP requests and responses is defined according to ISO 15118-2:2014, 7.10.1.

7.7.1.5 V2G messages

The TTCN-3 data representation of all V2G messages defined in ISO 15118-2:2014, Clause 8, is handled according to ETSI ES 201 873-9 V4.6.1 (TTCN-3: Using XML schema with TTCN-3). It defines an unambiguous mapping between XML Schema based type definitions and a corresponding TTCN-3 data type representation. The resulting TTCN-3 data structures for testing V2G messaging conformance are defined in Annex O, O.6.

7.7.1.6 X.509 Certificates

The TTCN-3 data representation of X.509 Certificates is defined in addition to ISO 15118-2:2014, Clause 8 in XML schema and is handled according to ETSI ES 201 873-9 V4.6.1 (TTCN-3: Using XML schema with TTCN-3). It defines an unambiguous mapping between XML Schema based type definitions and a corresponding TTCN-3 data type representation. The resulting TTCN-3 data structures for testing V2G messaging conformance including X.509 Certificates are defined in Annex O, O.6.

7.7.2 Templates

A template defines at least one value for a data type. The templates in this document are used for two purposes:

- representation of stimulus test data; and
- SUT response matching with expected behavior.

The templates relevant for this document are defined in Annex J - Annex N. Templates that are only used once in a concrete test case are defined in their corresponding context in the Annexes. All templates that are used in several test cases / contexts are defined in the common sections in the Annexes.

The templates for the V2G messages defined in the XML Schema of ISO 15118-2 consist of a header (`MessageHeaderType`) and a body (`BodyElement_group`). There are templates that can be parameterized. At least one template for each message exists, where all elements of the first level of the message type are defined as parameters of the template.

In the TSS the following rules apply for templates used in the context of test cases for EVCC/SECC.

- For testing an SECC every attribute in an EVCC request message is either allocated or omitted.
- For testing an EVCC every attribute in an SECC response message is allocated either by parameters of the template or TTCN-3 wildcards.

7.7.3 Timeouts and timers

All timeouts are defined according to ISO 15118-2. For the scope of this document *performance times* in ISO 15118-2 are irrelevant since they are not visible to the tester (black box testing). Hence only *timeouts* can be exactly measured from the tester's perspective and are considered in this document. The resulting timeout definitions for the TTCN-3 test suite are provided in Annex A, A.1.

The timer handling in the test suite is defined as follows: Before sending a stimulus to the SUT the corresponding message timer is started. After receiving the expected response from the SUT the message timer will be stopped. In case no response is received from the SUT until the point in time defined by the corresponding timeout value, the test verdict is set to 'false'. The timer handling is modelled as part of the test behavior section within the test case specification.

7.7.4 Common behavior

Within the TSS few functions are defined which are used in different contexts thus being invoked from various callers. Such functions are defined in Annex F, F.4.

7.7.5 Test case modelling

One test case always executes a specific test behavior and verifies the SUT behavior according to its message based response. All test cases always begin and end in a defined and recoverable state of the SUT and the tester. As a consequence, every test case consists of the following phases.

- a) Initialize.
 - 1) Create PTC components.

- 2) Map all MTC/PTC/system ports and communication bindings according to the underlying test configuration.

b) Pre-condition

The test system is supposed to start in a state so that a communication or data exchange with the SUT is possible. In the pre-condition of a test case the tester as well as the SUT therefore initialize into known and stable state. This incorporates the following action.

- 1) Bring SUT and tester to a valid and known state before the actual test behavior is executed.

c) Test behavior

The actual test behavior defines the set of steps during the test execution which are essential in order to achieve the test purpose and assign verdicts to the possible outcomes. This involves the following actions.

- 1) Initialize and start relevant timer(s) on the tester side.
- 2) Execute test behavior (send stimulus to the SUT).
- 3) Listen to any events and verify SUT response (stop timers).
- 4) Assign test verdict for this test behavior.

d) Post-condition

- 1) Bring SUT and tester into final/initial state.

e) Shutdown

- 1) Unmap all connected ports & communication bindings.
- 2) Shutdown PTC components.

7.7.6 Message handling depending on SUT type

In reference to the test case modelling as defined in the previous subclause, a stimulus is always sent from the tester to the SUT as part of the test behavior (phase b2). Due to the strictly defined client, server roles assigned to the EVCC and SECC respectively, the following two message handling paradigms result depending on the type of the SUT.

- a) If the SUT is an SECC: The tester sends a request message as stimulus to the SUT during phase c2 (see 7.7.5) and expects the corresponding V2G response message back from the SUT.
- b) If the SUT is an EVCC: The tester sends a response message as stimulus to the SUT during phase c2 (see 7.7.5) and expects a V2G request message according to the next message pattern in the V2G message sequence back from the SUT. If multiple next messages are defined, the tester shall be able to handle these in order to allow for valid verdict assignments.

7.7.7 IEC 61851-1 PWM event handling and control

The parallel test component (PTC) for IEC 61851-1 signaling continuously observes the protocol specific parameters positive voltage (state), frequency, duty cycle and the proximity resistor value in accordance with the defined valid range of values (as defined in IEC 61851-1:2017, Annex A), during the entire test case execution.

Requirements of IEC 61851-1 SUT adapter for SECC or EVCC testing

- [V2G4-022] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a function to get the current PWM state or associated voltage level.
- [V2G4-023] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a call-back function which reports change events of the current PWM state or associated voltage level.
- [V2G4-024] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a function to set the proximity on SUT side.
- [V2G4-025] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a function to get the proximity resistor value from SUT side.
- [V2G4-026] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a call-back function which reports change events of the resistor value for the proximity from SUT side.
- [V2G4-027] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a function to get the duty cycle in [%].
- [V2G4-028] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a call-back function which reports changes of the duty cycle in [%].
- [V2G4-029] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a function to get the frequency in [Hz].
- [V2G4-030] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a call-back function which reports change events of the frequency in [Hz].

Requirements of IEC 61851-1 SUT adapter for SECC testing

- [V2G4-031] The IEC 61851-1 SUT adapter for SECC testing shall provide a function to set the current PWM state or associated voltage level. State B, C and D or the associated voltage level shall be supported, other voltage level are optional.

Requirements of IEC 61851-1 SUT adapter for EVCC testing

- [V2G4-032] The IEC 61851-1 SUT adapter for EVCC testing shall provide a function to set the current PWM state or associated voltage level. State E and F or the associated voltage level shall be supported, other voltage levels are optional.
- [V2G4-033] The IEC 61851-1 SUT adapter for EVCC testing shall provide a function to set the current duty cycle in [%]. The values 5 %, and 7 %-98 % shall be supported, other values are optional.
- [V2G4-034] The IEC 61851-1 SUT adapter for EVCC testing shall provide a function to set current frequency in [Hz]. A value of 1 000 Hz +/- 5 % shall be supported, other values are optional.
- [V2G4-035] The IEC 61851-1 SUT adapter for EVCC testing shall provide a function to switch on/off the ControlPilot Line to simulate an un-plug or plug-in for the SUT. State A or the associated voltage level shall be supported, other voltage level is optional.

Requirements of IEC 61851-1 PTC for SECC or EVCC testing

- [V2G4-036] The IEC 61851 PTC for SECC or EVCC testing shall provide a function which activates the validity monitoring as specified in [V2G4-036], [V2G4-037], [V2G4-038], [V2G4-039], [V2G4-040], [V2G4-041].
- [V2G4-037] The IEC 61851-1 PTC for SECC or EVCC testing shall provide a function to confirm a change from/to a given PWM state. This function shall wait for receiving a state change event from IEC 61851-1 SUT adapter. If no change event occurs within a given time [ms], the function shall return an error.
- [V2G4-038] The IEC 61851-1 PTC for SECC or EVCC testing shall provide a function to change its

valid PWM state condition. This function shall change the internal monitoring condition to a given PWM state, which the IEC 61851-1 PTC will consider as the next valid PWM state.

- [V2G4-039]** The IEC 61851-1 PTC for SECC or EVCC testing shall provide a function to change its valid duty cycle range. This function shall change the internal monitoring condition to a given duty cycle range from [%] – to [%], which the IEC 61851-1 PTC will consider as the next valid duty cycle range.

Requirements of IEC 61851-1 PTC for SECC testing

- [V2G4-040]** The IEC 61851-1 PTC for SECC testing shall provide a function to confirm a given duty cycle in [%]. This function shall wait for receiving a duty cycle change event from IEC 61851-1 SUT adapter. If no change event occurs within a given time [ms], the function shall report an error.
- [V2G4-041]** The IEC 61851-1 PTC for SECC testing shall provide a function to change its valid frequency range. This function shall change the internal monitoring condition to a given frequency range from [Hz] – to [Hz], which the IEC 61851-1 PTC will consider as the next frequency range.

7.7.8 Public Key Infrastructure (PKI) setup and certificate handling

The ISO 15118-2 requires the existence of multiple certificates to be applied for both the Transport Layer Security (TLS) and for XML-Security on application layer. SECC Certificates are used to unilaterally authenticate SECC's by EVCC'S and Contract Certificates are handled at the application layer to authenticate an EVCC against an SECC and/or secondary actor. V2G Root Certificates and possibly Sub-CA Certificates are used to certify SECC Certificates and EVCC Contract Certificates. For installing and updating Contract Certificates, there also may be OEM Root Certificates and OEM Provisioning Certificates used in addition.

Figure 4 provides a Schematic of the Public Key Infrastructure (PKI) setup used for testing of ISO 15118-2. In order to verify the SUT behavior not only with valid credentials, more than one PKI is required which is indicated in Figure 4 by #V2G-CAs, #MO-CAs, #OEM-CAs and #PE-CAs. Figure 4 also provides the number of Sub-CAs and Certificates (#CPO-Sub-CA, #Prov-Sub-CA, #MO-Sub-CA, #OEM-Sub-CA etc.) for every certificate chain. Not all permutations of these parameter are valid in terms of ISO 15118-2 requirements and are even not necessary at all. For reproducibility, Table 35 to Table 41 define the minimal PKI Configuration required for ISO 15118-2 testing. Since PKI-Tooling and the PKI-Timing depending on the overall test architecture, a specific setup or even a specific certificate set are not part of this document.

These credentials are handled as PICS/PIXIT. Each certificate also implies the existence of the associated private and public key as well as a certificate chain if required. Table 32, Table 33 and Table 34 define the ISO 15118 test suite naming convention for the PKI and certificate identifiers. Furthermore, some test case will also modify these PICS/PIXIT in order to verify the SUT behavior using *manipulated credentials*. The PICS/ PIXITs shall be available before test case execution. The process to provide these Information to the SUT is out of scope.

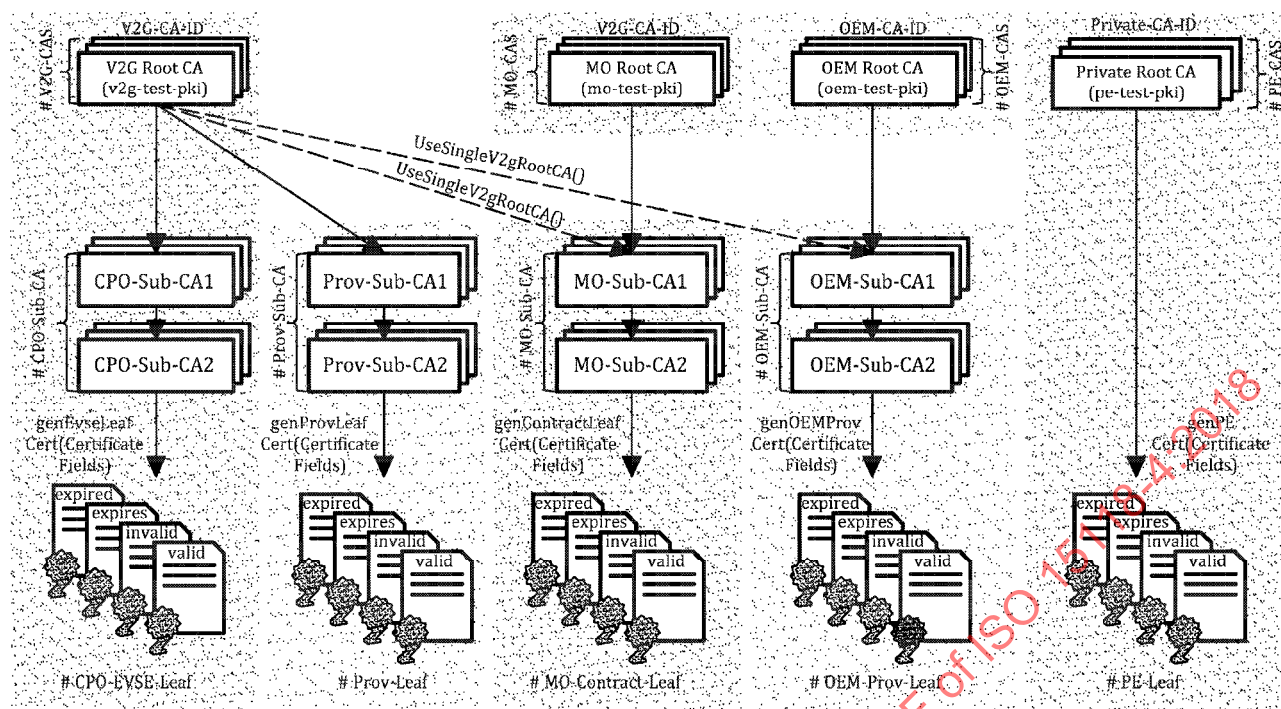


Figure 4 — Schematic Public Key Infrastructure Setup for testing of ISO 15118-2

Table 32 — ISO 15118 test suite naming convention for PKI identifiers

Identifier	Values	Description
<prefix>- <root_ca_nn>	PKI-{xxx}	The prefix for the Public Key Infrastructure (PKI) identifiers Number of the root CA certificate (#V2G-CAs) NOTE Every V2G root CA defines a new instance of the PKI.

Table 33 — ISO 15118 test suite naming convention for certificate identifiers

Identifier	Values	Description
<pki_name>	{pki_name}	See Table 32
<prefix>	CRT	The prefix for certificate identifiers
<cert_dom>		Domain the certificate belongs to
	V2G	If the certificate is issued for a V2G PKI Service
	CPO	If the certificate is issued for a Charge Point Operator
	PROV	If the certificate is issued for a Provisioning Service
	MO	If the certificate is issued for a Mobility Operator
	OEM	If the certificate is issued for a OEM
	PE	If the certificate is issued for a Private Environment
	EVSE	If the certificate is issued for a EVSE by a CPO
	CONTRACT	If the certificate is issued for a contract by MO
<cert_hierarchy>		Hierarchy level of the certificate
	ROOT	If the certificate is a root CA (CA == true)
	SUB1	If the certificate is a CA (CA == true) and (PathLength == 1)
	SUB2	If the certificate is a CA (CA == true) and (PathLength == 0)

	LEAF	If the certificate is a LEAF (CA == false)
<cert_type>		Type of the certificate
	VALID	If the certificate is valid and the time of expire will not influent the test process
	EXPIRED	If the certificate is expired before test case execution
	EXPIRES	If the certificate expires soon. The time limit shall be sufficient that it will trigger the certificate update procedure.
	INVALID_CONTENT	If any critical content (see ISO 15118 Annex F) is invalid
	INVALID_EMAID	If the emaid has an invalid syntax
	WRONG_EMAID	If the emaid differs from the original version when the contract was concluded (Syntax is correct)
	EXPIRED_ISSUER_SUB1	If the leaf certificate was derived from expired SUB 1 certificate
	EXPIRED_ISSUER_SUB2	If the leaf certificate was derived from expired SUB 2 certificate

Table 34 — ISO 15118 test suite naming convention for wildcards in certificate identifiers

Identifier	Values	Description
<wildcard>	*	The wildcard prefix for certificate identifiers is used in the PKI definition to abbreviate the pki- and certificate name until the <cert_type> in a given context.
<cert_type>		(See Table 37)

Table 35 — ISO 15118 parameters for Test PKI-1 (Multiple Root CAs)

PKI Name	PKI-1	
PKI Parameter Name	PKI Parameter Value	Certificate Name
#V2G-CAs/V2G-CA-ID	1/1	PKI-1_CRT_V2G_ROOT_VALID
#MO-CAs	1	PKI-1_CRT_MO_ROOT_VALID
#OEM-CAs	1	PKI-1_CRT_OEM_ROOT_VALID
#PE-CAs	0	—
#CPO-Sub-CA	2	PKI-1_CRT_CPO_SUB1_VALID, PKI-1_CRT_CPO_SUB2_VALID
#Prov-Sub-CA	2	PKI-1_CRT_PROV_SUB1_VALID, PKI-1_CRT_PROV_SUB2_VALID
#MO-Sub-CA	5	PKI-1_CRT_MO_SUB1_VALID, *_EXPIRED PKI-1_CRT_MO_SUB2_VALID *_EXPIRED, *_EXPIRED_ISSUER_SUB1
#OEM-Sub-CA	2	PKI-1_CRT_OEM_SUB1_VALID, PKI-1_CRT_OEM_SUB2_VALID
#CPO-EVSE-Leaf	2	PKI-1_CRT_EVSE_LEAF_VALID, *_EXPIRED
#Prov-Leaf	2	PKI-1_CRT_PROV_LEAF_VALID, *_EXPIRED
#MO-Contract-Leaf	8	PKI-1_CRT_CONTRACT_LEAF_VALID,

		* _ EXPIRED, * _ EXPIRES, * _ INVALID_CONTENT, * _ INVALID_EMAID, * _ WRONG_EMAID, * _ EXPIRED_ISSUER_SUB1, * _ EXPIRED_ISSUER_SUB2
#OEM-Prov-Leaf	3	PKI-1_CRT_OEM_LEAF_VALID, * _ EXPIRED, * _ INVALID_CONTENT
#PE_Leaf	0	—

Table 36 — ISO 15118 parameters for Test PKI-2 (Single V2G Root CA)

PKI Name	PKI-2	
PKI Parameter Name	PKI Parameter Value	Certificate Name
#V2G-CAs/V2G-CA-ID	1/2	PKI-2_CRT_V2G_ROOT_VALID
#MO-CAs	0	—
#OEM-CAs	0	—
#PE-CAs	0	—
#CPO-Sub-CA	2	PKI-2_CRT_CPO_SUB1_VALID, PKI-2_CRT_CPO_SUB2_VALID
#Prov-Sub-CA	2	PKI-2_CRT_PROV_SUB1_VALID, PKI-2_CRT_PROV_SUB2_VALID
#MO-Sub-CA	5	PKI-2_CRT_MO_SUB1_VALID, * _ EXPIRED PKI-2_CRT_MO_SUB2_VALID * _ EXPIRED, * _ EXPIRED_ISSUER_SUB1
#OEM-Sub-CA	2	PKI-2_CRT_OEM_SUB1_VALID, PKI-2_CRT_OEM_SUB2_VALID
#CPO-EVSE-Leaf	2	PKI-2_CRT_EVSE_LEAF_VALID, * _ EXPIRED
#Prov-Leaf	2	PKI-2_CRT_PROV_LEAF_VALID, * _ EXPIRED
#MO-Contract-Leaf	8	PKI-2_CRT_CONTRACT_LEAF_VALID, * _ EXPIRED, * _ EXPIRES, * _ INVALID_CONTENT, * _ INVALID_EMAID, * _ WRONG_EMAID, * _ EXPIRED_ISSUER_SUB1, * _ EXPIRED_ISSUER_SUB2
#OEM-Prov-Leaf	3	PKI-2_CRT_OEM_LEAF_VALID, * _ EXPIRED, * _ INVALID_CONTENT
#PE_Leaf	0	—

Table 37 — ISO 15118 parameters for Test PKI-3
(Multiple valid Root CA Certificates overlapping periods)

PKI Name	PKI-3	
PKI Parameter Name	PKI Parameter Value	Certificate Name
#V2G-CAs/V2G-CA-ID	1/3	PKI-3_CRT_V2G_ROOT_VALID_1, * _ VALID_2, * _ VALID_3, * _ VALID_4, * _ VALID_5, * _ VALID_6, * _ VALID_7, * _ VALID_8, * _ VALID_9, * _ VALID_10
#MO-CAs	1	PKI-3_CRT_MO_ROOT_VALID

#OEM-CAs	1	PKI-3_CRT_OEM_ROOT_VALID
#PE-CAs	0	—
#CPO-Sub-CA	2	PKI-3_CRT_CPO_SUB1_VALID, PKI-3_CRT_CPO_SUB2_VALID,
#Prov-Sub-CA	2	PKI-3_CRT_PROV_SUB1_VALID, PKI-3_CRT_PROV_SUB2_VALID,
#MO-Sub-CA	2	PKI-3_CRT_MO_SUB1_VALID, PKI-3_CRT_MO_SUB2_VALID,
#OEM-Sub-CA	2	PKI-3_CRT_OEM_SUB1_VALID, PKI-3_CRT_OEM_SUB2_VALID,
#CPO-EVSE-Leaf	1	PKI-3_CRT_EVSE_LEAF_VALID
#Prov-Leaf	1	PKI-3_CRT_PROV_LEAF_VALID,
#MO-Contract-Leaf	1	PKI-3_CRT_CONTRACT_LEAF_VALID
#OEM-Prov-Leaf	1	PKI-3_CRT_OEM_LEAF_VALID
#PE_Leaf	0	—

**Table 38 — ISO 15118 parameters for Test PKI-4
(Multiple valid Root CA Certificates single SUB-CAs)**

PKI Name	PKI-4	
PKI Parameter Name	PKI Parameter Value	Certificate Name
#V2G-CAs/V2G-CA-ID	1/4	PKI-4_CRT_V2G_ROOT_VALID
#MO-CAs	1	PKI-4_CRT_MO_ROOT_VALID
#OEM-CAs	1	PKI-4_CRT_OEM_ROOT_VALID
#PE-CAs	0	—
#CPO-Sub-CA	1	PKI-4_CRT_CPO_SUB1_VALID
#Prov-Sub-CA	1	PKI-4_CRT_PROV_SUB1_VALID
#MO-Sub-CA	1	PKI-4_CRT_MO_SUB1_VALID
#OEM-Sub-CA	1	PKI-4_CRT_OEM_SUB1_VALID
#CPO-EVSE-Leaf	1	PKI-4_CRT_EVSE_LEAF_VALID
#Prov-Leaf	1	PKI-4_CRT_PROV_LEAF_VALID
#MO-Contract-Leaf	1	PKI-4_CRT_CONTRACT_LEAF_VALID
#OEM-Prov-Leaf	1	PKI-4_CRT_OEM_LEAF_VALID
#PE_Leaf	0	—

**Table 39 — ISO 15118 parameters for Test PKI-5
(Path Length >=4 AND Leaf Certificate issued by Root)**

PKI Name	PKI-5	
PKI Parameter Name	PKI Parameter Value	Certificate Name
#V2G-CAs/V2G-CA-ID	1/5	PKI-5_CRT_V2G_ROOT_VALID, PKI-5_CRT_CONTRACT_LEAF_ISSUER_IS_ROOT
#MO-CAs	1	PKI-5_CRT_MO_ROOT_VALID

#OEM-CAs	1	PKI-5_CRT_OEM_ROOT_VALID
#PE-CAs	0	—
#CPO-Sub-CA	3	PKI-5_CRT_CPO_SUB1_VALID, PKI-5_CRT_CPO_SUB2_VALID, PKI-5_CRT_CPO_SUB3_VALID
#Prov-Sub-CA	3	PKI-5_CRT_PROV_SUB1_VALID, PKI-5_CRT_PROV_SUB2_VALID, PKI-5_CRT_PROV_SUB3_VALID
#MO-Sub-CA	3	PKI-5_CRT_MO_SUB1_VALID, PKI-5_CRT_MO_SUB2_VALID, PKI-5_CRT_MO_SUB3_VALID
#OEM-Sub-CA	3	PKI-5_CRT_OEM_SUB1_VALID, PKI-5_CRT_OEM_SUB2_VALID, PKI-5_CRT_OEM_SUB3_VALID
#CPO-EVSE-Leaf	1	PKI-5_CRT_EVSE_LEAF_VALID
#Prov-Leaf	1	PKI-5_CRT_PROV_LEAF_VALID
#MO-Contract-Leaf	1	PKI-5_CRT_CONTRACT_LEAF_VALID
#OEM-Prov-Leaf	1	PKI-5_CRT_OEM_LEAF_VALID
#PE_Leaf	0	—

Table 40 — ISO 15118 parameters for Test PKI-6 (Private Environment (PE))

PKI Name	PKI-6	
PKI Parameter Name	PKI Parameter Value	Certificate Name
#V2G-CAs/V2G-CA-ID	0	—
#MO-CAs	0	—
#OEM-CAs	0	—
#PE-CAs/PE-CA-ID	1/6	PKI-6_CRT_PE_ROOT_VALID
#CPO-Sub-CA	0	—
#Prov-Sub-CA	0	—
#MO-Sub-CA	0	—
#OEM-Sub-CA	0	—
#CPO-EVSE-Leaf	0	—
#Prov-Leaf	0	—
#MO-Contract-Leaf	0	—
#OEM-Prov-Leaf	0	—
#PE_Leaf	3	PKI-6_CRT_OEM_LEAF_VALID, *_ EXPIRED, *_ INVALID_CONTENT

Table 41 — ISO 15118 parameters for Test PKI-7 (PE Root CA expires)

PKI Name	PKI-8	
PKI Parameter Name	PKI Parameter Value	Certificate Name

#V2G-CAs/V2G-CA-ID	0	—
#MO-CAs	0	—
#OEM-CAs	0	—
#PE-CAs/PE-CA-ID	1/7	PKI-7_CRT_PE_ROOT_EXPIRES
#CPO-Sub-CA	0	—
#Prov-Sub-CA	0	—
#MO-Sub-CA	0	—
#OEM-Sub-CA	0	—
#CPO-EVSE-Leaf	0	—
#Prov-Leaf	0	—
#MO-Contract-Leaf	0	—
#OEM-Prov-Leaf	0	—
#PE_Leaf	1	PKI-7_CRT_PE_LEAF_VALID

[V2G4-042] The Test-PKI shall consider all ISO 15118-2 requirements which are summarized in Table 42, in order to setup the valid certificates and corresponding certificate chains.

Table 42 — ISO 15118-2 requirements for valid PKI setup

ISO 15118-2 PKI Requirement	[V2G2-004], [V2G2-009], [V2G2-010], [V2G2-011], [V2G2-012], [V2G2-878], [V2G2-878], [V2G2-867], [V2G2-869], [V2G2-911], [V2G2-868], [V2G2-932], [V2G2-933]

[V2G4-043] At least one PKI shall have a path length ≥ 4 constraint of the PKI certificate tree.

[V2G4-044] At least one PKI shall have a 10 number of concurrently valid V2G Root Certificates for one Root CA. The certificate shall have the suffix *_VALID_1 to *_VALID_10.

7.7.9 Transport Layer Security (TLS)

The security concept of ISO 15118-2:2014, 7.3, provides a basic transport based protection mechanism based on Transport Layer Security (TLS). Hence the *TLS Communication Setup Handling* using certificates (7.7.8) shall be considered by the ATS. Since the verification of the TLS implementation itself as well as associated Cipher Suites are out of scope, the test cases focus only the *TLS Communication Setup*.

In order to test the setup of a protected TCP/IP based communication using a unilaterally authenticated TLS channel between EVCC and SECC, an EVCC_Tester or SECC_Tester needs a set of certificates, keys and certificate chains. These credentials are handled as PICS/PIXIT and are summarized based on PKI-1 in Table 43 and Table 44. Furthermore, some test case will also modify these PICS/PIXIT in order to verify the SUT behavior using *manipulated credentials*. The PICS/PIXITs shall be available before test case execution. The process to provide these Information to the SUT is out of scope.

Table 43 — Security Credentials EVCC_Tester TLS Communication Setup based on PKI-1

Certificates	Keys	Certificate Chain
PKI-1_CRT_EVSE_LEAF_VALID, *_EXPIRED	PKI-1_CRT_ EVSE_LEAF_*_ Private_Key	PKI-1_CRT_CPO_SUB1_VALID -> PKI-1_CRT_CPO_SUB2_VALID
PKI-1_CRT_EVSE_LEAF_VALID	PKI-1_CRT_ EVSE_LEAF_VALID_ Private_Key	PKI-1_CRT_CPO_SUB2_VALID -> PKI-1_CRT_CPO_SUB1_VALID (Invalid chain order)

Table 44 — Security Credentials SECC_Tester TLS Communication Setup based on PKI-1

Certificates	Keys	Certificate Chain
PKI-1_CRT_ V2G_ROOT_VALID	PKI-1_CRT_ EVSE_LEAF_VALID_ Public_Key	PKI-1_CRT_CPO_SUB1_VALID -> PKI-1_CRT_CPO_SUB2_VALID
PKI-1_CRT_ V2G_ROOT_VALID	PKI-1_CRT_ EVSE_LEAF_VALID_ Public_Key	PKI-1_CRT_CPO_SUB2_VALID -> PKI-1_CRT_CPO_SUB1_VALID (Invalid chain order)

Beside the PICSS/PIXITs additional information e.g. used cipher suite or handshake results are required, to handle the *TLS Communication Setup*. These data are structured in Annex O, O.3. The described data structures are used to setup and control the TLS communication session.

The setup information will be used by the port map operation to configure the TLS connection.

The control information will be sent to the TLS port using the *sent to* directive with destination is set to *local*. This enables to set port control information or get TLS handshake information.

7.7.10 XML Security

Specific message content is signed on application layer (XML-messages), if data has to be protected on the way from or to a secondary actor, or if the integrity protection has to last longer than the existence of the TLS channel. To test these integrity protection means on application layer, additional external functions are required.

Signing and Verification requirements for XML security

- [V2G4-045]** The external function for XML security shall provide a mean to calculate the message digest using a given digest algorithm and data stream.
- [V2G4-046]** The external function for XML security shall provide a mean to encode XML data as EXI representation of this stream.
- [V2G4-047]** The external function for XML security shall provide a mean to sign a data stream based on a signature algorithm.
- [V2G4-048]** The external function for XML security shall provide a mean to verify a signed data stream. This function shall use a given signature algorithm, a signature, a leaf certificate and the EXI encoded data that should be signed. It shall return whether verification was successful or otherwise.
- [V2G4-049]** The external function for XML security shall provide a mean to verify a certificate chain up to the root CA. This function shall use a given certificate chain and a root CA for verification. It shall return whether verification was successful or otherwise.
- [V2G4-050]** The external function for XML security shall provide a mean to generate a GenChallenge value.

ECDH requirements for XML security

- [V2G4-051]** The external function for XML security shall provide a mean to calculate a shared secret based on ECDH. This function shall use a given certificate and a Diffie-Hellman public key.
- [V2G4-052]** The external function for XML security shall provide a mean to decrypt a private key based on ECDH. This function shall use a given shared secret and an encrypted private key.

Certificate handling requirements for XML security

- [V2G4-053]** The external function for XML security shall provide a mean to load a certificate encoded as X509CertificateType (refer to Annex O, O.6). This function shall use a given unique PKI and certificate name according to 7.7.8.

- [V2G4-054]** The external function for XML security shall provide a mean to load a private key related to a certificate. This function shall use a given unique certificate name according to clause 7.7.8.
- [V2G4-055]** The external function for XML security shall provide a mean to load the EMAID of the contract certificate.

SA requirements for XML security

- [V2G4-056]** The external function for XML security shall provide a mean to verify the authenticity of the OEMProvisioning leaf certificate by the SA. It shall return whether verification was successful or otherwise.
- [V2G4-057]** The external function for XML security shall provide a mean to update the contract certificate by the SA. It shall return whether verification was successful or otherwise.
- [V2G4-058]** The external function for XML security shall provide a mean to get the SAProvisioning certificate chain from the SA encoded as CertificateChainType (refer to Annex O, O.6). This function shall use a given OEMProvisioning leaf or contract certificate.
- [V2G4-059]** The external function for XML security shall provide a mean to get the contract certificate chain encoded from the SA as CertificateChainType (refer to Annex O, O.6). This function shall use a given OEMProvisioning leaf or contract certificate.
- [V2G4-060]** The external function for XML security shall provide a mean to get the contract signature encrypted private key from the SA encoded as ContractSignatureEncryptedPrivateKeyType (refer to Annex O, O.6). This function shall use a given OEMProvisioning leaf or contract certificate.
- [V2G4-061]** The external function for XML security shall provide a mean to get the Diffie-Hellman public key from the SA encoded as DiffieHellmanPublicKeyType (refer to Annex O, O.6). This function shall use a given OEMProvisioning leaf or contract certificate.
- [V2G4-062]** The external function for XML security shall provide a mean to get the EMAID of a contract certificate from the SA encoded as EMAIDType (refer to Annex O, O.6). This function shall use a given OEMProvisioning leaf or contract certificate.
- [V2G4-063]** The external function for XML security shall provide a mean to get the signature value for the signed message parts ContractSignatureCertChain, ContractSignatureEncryptedPrivateKey, DHpublickey and eMAID (refer to Annex O, O.6) from the SA. This function shall use a given OEMProvisioning leaf or contract certificate.

7.7.11 EIM status control functionality

In order to ensure EIM authorization, the MTC is able to detect and control EIM status information by using special functions depending on the MTC's type (SECC_Tester).

The parallel test component (PTC) for EIM status continuously observes the status of the EIM authorization until the authorization process is successful or failed.

Requirements of EIM status control for SECC testing

- [V2G4-064]** The service function for EIM status control shall provide a mean to initiate the EIM authorization.
- [V2G4-065]** The service function for EIM status control shall provide a mean to process incoming EIM status information from the EIM status PTC.

Requirements of EIM status PTC for SECC testing

- [V2G4-066]** The EIM status PTC for SECC testing shall provide a function to inform the MTC if the EIM authorization is successful. If no change event occurs within a given time [ms], the

function shall report an error.

8 Test case descriptions for 15118-2 V2GTP

8.1 General information

This clause covers all test case descriptions for the *V2GTP requirements* as defined in ISO 15118-2. Subclause 8.2 details all test case descriptions, where the *SECC* is the SUT. Subclause 8.3 details all test case descriptions, where the *EVCC* is the SUT.

8.2 SECC test cases

Table 45 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_001'.

Table 45 — Test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_001'

TC Id	TC_SECC_CMN_VTB_V2GTPSessionSetup_001
Test objective	Test System executes GoodCase procedure and sends a SessionSetupReq message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H and 'payloadType' equals to '0x8001'H. (V2GTP Header is matched for V2G message content). Test System then checks that the SUT sends a SessionSetupRes message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H and 'payloadType' equals to '0x8001'H.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-087], [V2G2-088]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_V2gtpV2g := true
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSessionSetup_001	

Table 46 — Test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_002'

Table 46 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_002'.

TC Id	TC_SECC_CMN_VTB_V2GTPSessionSetup_002
Test objective	Test System executes GoodCase procedure and sends an invalid 'payloadLength' equals to '00000000'H in the V2GTP header of the SessionSetupReq message. Test System then checks that the SUT ignores the SessionSetupReq message by detecting a 'V2G_Msg_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_V2gtpV2g := true

PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002	

Table 47 — Test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_003'

Table 47 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_003'.

TC Id	TC_SECC_CMN_VTB_V2GTPSessionSetup_003
Test objective	Test System executes GoodCase procedure and sends an invalid 'payloadType' equals to '7001'H in the V2GTP header of the SessionSetupReq message. Test System then checks that the SUT ignores the SessionSetupReq message by detecting a 'V2G_Msg_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_V2gtpV2g := true
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002	

Table 48 — Test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_004'

Table 48 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_004'.

TC Id	TC_SECC_CMN_VTB_V2GTPSessionSetup_004
Test objective	Test System executes GoodCase procedure and sends an invalid 'protocolVersion' equals to 'FF'H in the V2GTP header of the SessionSetupReq message. Test System then checks that the SUT ignores the SessionSetupReq message by detecting a 'V2G_Msg_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_V2gtpV2g := true
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002	

Table 49 — Test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_005'

Table 49 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_005'.

TC Id	TC_SECC_CMN_VTB_V2GTPSessionSetup_005
Test objective	Test System executes GoodCase procedure and sends an invalid 'invProtocolVersion' equals to 'FF'H in the V2GTP header of the SessionSetupReq message. Test System then checks that the SUT ignores the SessionSetupReq message by detecting a 'V2G_Msg_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_V2gtpV2g := true
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002	

Table 50 — Test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_001'

Table 50 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_001'.

TC Id	TC_SECC_CMN_VTB_V2GTPSDP_001
Test objective	Test System executes GoodCase procedure and sends a valid SDP request message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H and 'payloadType' equals to '0x9000'H. (V2GTP Header is matched for SDP message content). Test System then checks that the SUT sends a SDP response message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H, 'payloadType' equals to '0x9001'H and 'payloadLength' equals to '00000014'H.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.10.1.3.1; 15118-2:7.10.1.5
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-087], [V2G2-130], [V2G2-152], [V2G2-153]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_V2gtpSdp := true
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSDP_001	

Table 51 — Test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_002'

Table 51 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_002'.

TC Id	TC_SECC_CMN_VTB_V2GTPSDP_002

Test objective	Test System executes GoodCase procedure and sends an invalid 'payloadLength' equals to '00000000'H in the V2GTP header of the SDP request message. Test System then checks that the SUT ignores the SDP response message by detecting an SDP message timeout.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_V2gtpSdp := true
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSDP_002	

Table 52 — Test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_003'

Table 52 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_003'.

TC Id	TC_SECC_CMN_VTB_V2GTPSDP_003
Test objective	Test System executes GoodCase procedure and sends an invalid 'payloadType' equals to '7000'H in the V2GTP header of the SDP request message. Test System then checks that the SUT ignores the SDP response message by detecting an SDP message timeout.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_V2gtpSdp := true
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSDP_002	

Table 53 — Test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_004'

Table 53 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_004'.

TC Id	TC_SECC_CMN_VTB_V2GTPSDP_004
Test objective	Test System executes GoodCase procedure and sends an invalid 'protocolVersion' equals to 'FF'H in the V2GTP header of the SDP request message. Test System then checks that the SUT ignores the SDP response message by detecting an SDP message timeout.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2

Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_V2gtpSdp := true
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSDP_002	

Table 54 — Test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_005'

Table 54 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_005'.

TC Id	TC_SECC_CMN_VTB_V2GTPSDP_005
Test objective	Test System executes GoodCase procedure and sends an invalid 'invProtocolVersion' equals to 'FF'H in the V2GTP header of the SDP request message. Test System then checks that the SUT ignores the SDP response message by detecting an SDP message timeout.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_V2gtpSdp := true
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSDP_002	

8.3 EVCC test cases

Table 55 — Test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_001'

Table 55 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_001'.

TC Id	TC_EVCC_CMN_VTB_V2GTPSessionSetup_001
Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolRes message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H and 'payloadType' equals to '0x8001'H. (V2GTP Header is matched for V2G message content). Test System then checks that the SUT sends a SessionSetupReq message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H and 'payloadType' equals to '0x8001'H.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-087], [V2G2-088]

Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_V2gtpV2g := true
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_001	

Table 56 — Test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_002'

Table 56 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_002'.

TC Id	TC_EVCC_CMN_VTB_V2GTPSessionSetup_002
Test objective	Test System executes GoodCase procedure and sends an invalid 'payloadLength' equals to '00000000'H in the V2GTP header of the SessionSetupRes message. Test System then checks that the SUT ignores the SessionSetupRes message by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if the 'V2G_Msg_Timeout' timer has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_V2gtpV2g := true
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002	

Table 57 — Test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_003'

Table 57 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_003'.

TC Id	TC_EVCC_CMN_VTB_V2GTPSessionSetup_003
Test objective	Test System executes GoodCase procedure and sends an invalid 'payloadType' equals to '7001'H in the V2GTP header of the SessionSetupRes message. Test System then checks that the SUT ignores the SessionSetupRes message by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if the 'V2G_Msg_Timeout' timer has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_V2gtpV2g := true
PIXIT selection	
PreCondition	

f_EVCC_CMN_PR_SessionSetup_001
Expected behavior
f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002

Table 58 — Test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_004'

Table 58 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_004'.

TC Id	TC_EVCC_CMN_VTB_V2GTPSessionSetup_004
Test objective	Test System executes GoodCase procedure and sends an invalid 'protocolVersion' equals to 'FF'H in the V2GTP header of the SessionSetupRes message. Test System then checks that the SUT ignores the SessionSetupRes message by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if the 'V2G_Msg_Timeout' timer has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_V2gtpV2g := true
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002	

Table 59 — Test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_005'

Table 59 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_005'.

TC Id	TC_EVCC_CMN_VTB_V2GTPSessionSetup_005
Test objective	Test System executes GoodCase procedure and sends an invalid 'invProtocolVersion' equals to 'FF'H in the V2GTP header of the SessionSetupRes message. Test System then checks that the SUT ignores the SessionSetupRes message by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if the 'V2G_Msg_Timeout' timer has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_V2gtpV2g := true
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002	

Table 60 — Test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_001'

Table 60 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_001'.

TC Id	TC_EVCC_CMN_VTB_V2GTPSDP_001
Test objective	Test System executes GoodCase procedure, starts SDP server and waits for an SDP request message (V2GTP Header is matched for SDP content). Test System then checks that the SUT sends a SDP request message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H, 'payloadType' equals to '0x9000'H and 'payloadLength' equals to 2.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.10.1.3.1; 15118-2:7.10.1.4
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-087], [V2G2-127], [V2G2-140], [V2G2-141]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_V2gtpSdp := true
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSDP_001	

Table 61 — Test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_002'

Table 61 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_002'.

TC Id	TC_EVCC_CMN_VTB_V2GTPSDP_002
Test objective	Test System executes GoodCase procedure and waits for a valid SDP request message. Test System then sends an invalid 'payloadLength' equals to '00000000'H in the V2GTP header of the SDP response message. Test System then checks the repetition of SDP request message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H, 'payloadType' equals to '0x9000'H and 'payloadLength' equals to 2.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_V2gtpSdp := true
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSDP_002	

Table 62 — Test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_003'

Table 62 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_003'.

TC Id	TC_EVCC_CMN_VTB_V2GTPSDP_003
Test objective	Test System executes GoodCase procedure and waits for a valid SDP request message.

	Test System then sends an invalid 'payloadType' equals to '7001'H in the V2GTP header of the SDP response message. Test System then checks the repetition of SDP request message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H, 'payloadType' equals to '0x9000'H and 'payloadLength' equals to 2.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_V2gtpSdp := true
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSDP_002	

Table 63 — Test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_004'

Table 63 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_004'.

TC Id	TC_EVCC_CMN_VTB_V2GTPSDP_004
Test objective	Test System executes GoodCase procedure and waits for a valid SDP request message. Test System then sends an invalid 'protocolVersion' equals to 'FF'H in the V2GTP header of the SDP response message. Test System then checks the repetition of SDP request message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H, 'payloadType' equals to '0x9000'H and 'payloadLength' equals to 2.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_V2gtpSdp := true
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSDP_002	

Table 64 — Test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_005'

Table 64 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_005'.

TC Id	TC_EVCC_CMN_VTB_V2GTPSDP_005
Test objective	Test System executes GoodCase procedure and waits for a valid SDP request message. Test System then sends an invalid 'invProtocolVersion' equals to 'FF'H in the V2GTP header of the SDP response message. Test System then checks the repetition of SDP request message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H,

	'payloadType' equals to '0x9000'H and 'payloadLength' equals to 2.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.8.3.1; 15118-2:7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_V2gtpSdp := true
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSDP_002	

9 Test case descriptions for 15118-2 SDP messages

9.1 General information

This clause covers all test case descriptions for the *SDP requirements* as defined in ISO 15118-2. Subclause 9.2 details all test case descriptions, where the *SECC* is the SUT. Subclause 9.3 details all test case descriptions, where the *EVCC* is the SUT.

9.2 SECC test cases

Table 65 — Test case description for 'TC_SECC_CMN_VTB_SDP_001'

Table 65 lists the test case description for 'TC_SECC_CMN_VTB_SDP_001'.

TC Id	TC_SECC_CMN_VTB_SDP_001
Test objective	Test System executes GoodCase procedure and sends an SDP request message with 'Security' equals to '0x10'H and 'TransportProtocol' equals to '0x00'H. Test System then checks that the SUT sends an SDP response message with 'Security' equals to '0x10'H, 'TransportProtocol' equals to '0x00'H and a valid port and IP address.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.1; 15118-2:7.10.1.5; 15118-2:7.10.1.7; 15118-2:7.4; 15118-2:8.7.4.4; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-123], [V2G2-147], [V2G2-150], [V2G2-154], [V2G2-624], [V2G2-625], [V2G2-027], [V2G2-144], [V2G2-915], [V2G2-852]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_TLS := false
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SDP_001	

Table 66 — Test case description for 'TC_SECC_CMN_VTB_SDP_002'

Table 66 lists the test case description for 'TC_SECC_CMN_VTB_SDP_002'.

TC Id	TC_SECC_CMN_VTB_SDP_002
Test objective	Test System executes GoodCase procedure and sends an SDP request message with 'Security' equals to '0x00'H and 'TransportProtocol' equals to '0x00'H. Test System then checks that the SUT sends an SDP response message with 'Security' equals to '0x00'H, 'TransportProtocol' equals to '0x00'H and a valid port and IP address.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.1; 15118-2:7.10.1.5; 15118-2:7.10.1.7; 15118-2:7.4; 15118-2:8.7.4.4; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-123], [V2G2-147], [V2G2-150], [V2G2-154], [V2G2-624], [V2G2-625], [V2G2-027], [V2G2-144], [V2G2-915], [V2G2-626], [V2G2-852]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM

PIXIT selection	PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SDP_001	

Table 67 — Test case description for 'TC_SECC_CMN_VTB_SDP_003'

Table 67 lists the test case description for 'TC_SECC_CMN_VTB_SDP_003'.

TC Id	TC_SECC_CMN_VTB_SDP_003
Test objective	Test System executes GoodCase procedure and sends an SDP request message with 'Security' equals to '0x00'H and 'TransportProtocol' equals to '0x00'H. Test System then checks that the SUT sends an SDP response message with 'Security' equals to '0x00'H, 'TransportProtocol' equals to '0x00'H and a valid port and IP address.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.1; 15118-2:7.10.1.5; 15118-2:7.10.1.7; 15118-2:7.4; 15118-2:7.3.4; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-123], [V2G2-147], [V2G2-150], [V2G2-154], [V2G2-624], [V2G2-626], [V2G2-027], [V2G2-144], [V2G2-631], [V2G2-915]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SDP_001	

Table 68 — Test case description for 'TC_SECC_CMN_VTB_SDP_004'

Table 68 lists the test case description for 'TC_SECC_CMN_VTB_SDP_004'.

TC Id	TC_SECC_CMN_VTB_SDP_004
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_SECC_CommunicationSetup_Timeout' timer has expired on SUT side. Test System then checks that the SUT stops the SDP server and does not respond to a valid SDP request message with 'Security' equals to '0x10'H and 'TransportProtocol' equals to '0x00'H.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.7.3.1
Referenced requirement(s)	[V2G2-723], [V2G2-029], [V2G2-605]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_TLS := false
PreCondition	
f_SECC_CMN_PR_StateB_001	

Expected behavior
f_SECC_CMN_TB_VTB_SDP_002

Table 69 — Test case description for 'TC_SECC_CMN_VTB_SDP_005'

Table 69 lists the test case description for 'TC_SECC_CMN_VTB_SDP_005'.

TC Id	TC_SECC_CMN_VTB_SDP_005
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_SECC_CommunicationSetup_Timeout' timer has expired on SUT side. Test System then checks that the SUT stops the SDP server and does not respond to a valid SDP request message with 'Security' equals to '0x00'H and 'TransportProtocol' equals to '0x00'H.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.7.3.1
Referenced requirement(s)	[V2G2-723], [V2G2-029], [V2G2-605]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SDP_002	

Table 70 — Test case description for 'TC_SECC_CMN_VTB_SDP_006'

Table 70 lists the test case description for 'TC_SECC_CMN_VTB_SDP_006'.

TC Id	TC_SECC_CMN_VTB_SDP_006
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_SECC_CommunicationSetup_Timeout' timer has expired on SUT side. Test System then checks that the SUT stops the SDP server and does not respond to a valid SDP request message with 'Security' equals to '0x00'H and 'TransportProtocol' equals to '0x00'H.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.7.3.1
Referenced requirement(s)	[V2G2-723], [V2G2-029], [V2G2-605]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SDP_002	

Table 71 — Test case description for 'TC_SECC_CMN_VTB_SDP_007'

Table 71 lists the test case description for 'TC_SECC_CMN_VTB_SDP_007'.

TC Id	TC_SECC_CMN_VTB_SDP_007
Test objective	Test System executes GoodCase procedure and sends an SDP request message with 'Security' equals to '0x00'H and 'TransportProtocol' equals to '0x00'H. Test System then checks that the SUT sends an SDP response message with 'Security' equals to '0x10'H, 'TransportProtocol' equals to '0x00'H and a valid port and IP address.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.1; 15118-2:7.10.1.5; 15118-2:7.10.1.7; 15118-2:7.4; 15118-2:8.7.4.4; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-123], [V2G2-147], [V2G2-150], [V2G2-154], [V2G2-624], [V2G2-027], [V2G2-144], [V2G2-915], [V2G2-627], [V2G2-852]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_TLS := false
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SDP_003	

9.3 EVCC test cases

9.3.1 Common test cases

Table 72 — Test case description for 'TC_EVCC_CMN_VTB_SDP_001'

Table 72 lists the test case description for 'TC_EVCC_CMN_VTB_SDP_001'.

TC Id	TC_EVCC_CMN_VTB_SDP_001
Test objective	Test System executes GoodCase procedure, starts SDP server and waits for an SDP request message. Test System then checks that the SUT sends an SDP request message with 'Security' equals to '0x10' and 'TransportProtocol' equals to '0x00'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.4
Referenced requirement(s)	[V2G2-142], [V2G2-018], [V2G2-019], [V2G2-623]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_001	

Table 73 — Test case description for 'TC_EVCC_CMN_VTB_SDP_002'

Table 73 lists the test case description for 'TC_EVCC_CMN_VTB_SDP_002'.

TC Id	TC_EVCC_CMN_VTB_SDP_002

Test objective	Test System executes GoodCase procedure, starts SDP server and waits for an SDP request message. Test System then checks that the SUT sends an SDP request message with 'Security' equals to '0x00' and 'TransportProtocol' equals to '0x00'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.4
Referenced requirement(s)	[V2G2-142], [V2G2-018], [V2G2-019], [V2G2-623]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_001	

Table 74 — Test case description for 'TC_EVCC_CMN_VTB_SDP_003'

Table 74 lists the test case description for 'TC_EVCC_CMN_VTB_SDP_003'.

TC Id	TC_EVCC_CMN_VTB_SDP_003
Test objective	Test System executes GoodCase procedure, starts SDP server and waits for an SDP request message. Test System then checks that the SUT sends an SDP request message with a valid security option and 'TransportProtocol' equals to '0x00'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.4
Referenced requirement(s)	[V2G2-142], [V2G2-018], [V2G2-019], [V2G2-623]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_001	

Table 75 — Test case description for 'TC_EVCC_CMN_VTB_SDP_004'

Table 75 lists the test case description for 'TC_EVCC_CMN_VTB_SDP_004'.

TC Id	TC_EVCC_CMN_VTB_SDP_004
Test objective	Test System executes GoodCase procedure, starts SDP server and waits for an SDP request message. Test System then checks that the SUT sends an SDP request message with 'Security' '0x00' and 'TransportProtocol' equals to '0x00'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.4; 15118-2:7.3.4

Referenced requirement(s)	[V2G2-142], [V2G2-018], [V2G2-019], [V2G2-623], [V2G2-630]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_001	

Table 76 — Test case description for 'TC_EVCC_CMN_VTB_SDP_005'

Table 76 lists the test case description for 'TC_EVCC_CMN_VTB_SDP_005'.

TC Id	TC_EVCC_CMN_VTB_SDP_005
Test objective	Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 77 — Test case description for 'TC_EVCC_CMN_VTB_SDP_006'

Table 77 lists the test case description for 'TC_EVCC_CMN_VTB_SDP_006'.

TC Id	TC_EVCC_CMN_VTB_SDP_006
Test objective	Test System executes GoodCase procedure, starts SDP server and waits for at least two SDP request messages and sends asynchronously two SDP response messages. Test System then checks if the SUT is able to handle an asynchronously arriving SECC Discovery Response message by waiting for a valid SupportedAppProtocolReq on the desired TCP link.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.4; 15118:7.10.1.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-018], [V2G2-019], [V2G2-126], [V2G2-159]
Config Id	CF_04_002
PICS selection	
PIXIT selection	

PreCondition
f_EVCC_CMN_PR_StateB_001
Expected behavior
f_EVCC_CMN_TB_VTB_SDP_003

Table 78 — Test case description for 'TC_EVCC_CMN_VTB_SDP_007'

Table 78 lists the test case description for 'TC_EVCC_CMN_VTB_SDP_007'.

TC Id	TC_EVCC_CMN_VTB_SDP_007
Test objective	Test System executes GoodCase procedure and sends an SDP request message with a valid security option and 'TransportProtocol' equals to '0x00'. Test System then checks that the SUT does not respond to an SDP request message from the SDP server.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.5
Referenced requirement(s)	[V2G2-145]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_004	

9.3.2 AC specific test cases

Table 79 — Test case description for 'TC_EVCC_AC_VTB_SDP_001'

Table 79 lists the test case description for 'TC_EVCC_AC_VTB_SDP_001'.

TC Id	TC_EVCC_AC_VTB_SDP_001
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002

PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp> PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 80 — Test case description for 'TC_EVCC_AC_VTB_SDP_002'

Table 80 lists the test case description for 'TC_EVCC_AC_VTB_SDP_002'.

TC Id	TC_EVCC_AC_VTB_SDP_002
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 81 — Test case description for 'TC_EVCC_AC_VTB_SDP_003'

Table 81 lists the test case description for 'TC_EVCC_AC_VTB_SDP_003'.

TC Id	TC_EVCC_AC_VTB_SDP_003
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network

	by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 82 — Test case description for 'TC_EVCC_AC_VTB_SDP_004'

Table 82 lists the test case description for 'TC_EVCC_AC_VTB_SDP_004'.

TC Id	TC_EVCC_AC_VTB_SDP_004
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true

PIXIT selection	PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 83 — Test case description for 'TC_EVCC_AC_VTB_SDP_005'

Table 83 lists the test case description for 'TC_EVCC_AC_VTB_SDP_005'.

TC Id	TC_EVCC_AC_VTB_SDP_005
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 84 — Test case description for 'TC_EVCC_AC_VTB_SDP_006'

Table 84 lists the test case description for 'TC_EVCC_AC_VTB_SDP_006'.

TC Id	TC_EVCC_AC_VTB_SDP_006
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection

	(old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 85 — Test case description for 'TC_EVCC_AC_VTB_SDP_007'

Table 85 lists the test case description for 'TC_EVCC_AC_VTB_SDP_007'.

TC Id	TC_EVCC_AC_VTB_SDP_007
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	

f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001
Expected behavior
f_EVCC_CMN_TB_VTB_SDP_002

Table 86 — Test case description for 'TC_EVCC_AC_VTB_SDP_008'

Table 86 lists the test case description for 'TC_EVCC_AC_VTB_SDP_008'.

TC Id	TC_EVCC_AC_VTB_SDP_008
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 87 — Test case description for 'TC_EVCC_AC_VTB_SDP_009'

Table 87 lists the test case description for 'TC_EVCC_AC_VTB_SDP_009'.

TC Id	TC_EVCC_AC_VTB_SDP_009
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element=0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure,

	starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 88 — Test case description for 'TC_EVCC_AC_VTB_SDP_010'

Table 88 lists the test case description for 'TC_EVCC_AC_VTB_SDP_010'.

TC Id	TC_EVCC_AC_VTB_SDP_010
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element=0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true

PIXIT selection	PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 89 — Test case description for 'TC_EVCC_AC_VTB_SDP_011'

Table 89 lists the test case description for 'TC_EVCC_AC_VTB_SDP_011'.

TC Id	TC_EVCC_AC_VTB_SDP_011
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element=0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 90 — Test case description for 'TC_EVCC_AC_VTB_SDP_012'

Table 90 lists the test case description for 'TC_EVCC_AC_VTB_SDP_012'.

TC Id	TC_EVCC_AC_VTB_SDP_012
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element=0W (par_SECC_Pmax0W) within

	ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 91 — Test case description for 'TC_EVCC_AC_VTB_SDP_013'

Table 91 lists the test case description for 'TC_EVCC_AC_VTB_SDP_013'.

TC Id	TC_EVCC_AC_VTB_SDP_013
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element=0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002

PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 92 — Test case description for 'TC_EVCC_AC_VTB_SDP_014'

Table 92 lists the test case description for 'TC_EVCC_AC_VTB_SDP_014'.

TC Id	TC_EVCC_AC_VTB_SDP_014
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element=0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 93 — Test case description for 'TC_EVCC_AC_VTB_SDP_015'

Table 93 lists the test case description for 'TC_EVCC_AC_VTB_SDP_015'.

TC Id	TC_EVCC_AC_VTB_SDP_015
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element=0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 94 — Test case description for 'TC_EVCC_AC_VTB_SDP_016'

Table 94 lists the test case description for 'TC_EVCC_AC_VTB_SDP_016'.

TC Id	TC_EVCC_AC_VTB_SDP_016
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element=0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS

	Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

9.3.3 DC specific test cases

Table 95 — Test case description for 'TC_EVCC_DC_VTB_SDP_001'

Table 95 lists the test case description for 'TC_EVCC_DC_VTB_SDP_001'.

TC Id	TC_EVCC_DC_VTB_SDP_001
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5 % duty cycle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 96 — Test case description for 'TC_EVCC_DC_VTB_SDP_002'

Table 96 lists the test case description for 'TC_EVCC_DC_VTB_SDP_002'.

TC Id	TC_EVCC_DC_VTB_SDP_002
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5 % duty cycle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := tIs, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 97 — Test case description for 'TC_EVCC_DC_VTB_SDP_003'

Table 97 lists the test case description for 'TC_EVCC_DC_VTB_SDP_003'.

TC Id	TC_EVCC_DC_VTB_SDP_003
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5 % duty cycle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2

Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 98 — Test case description for 'TC_EVCC_DC_VTB_SDP_004'

Table 98 lists the test case description for 'TC_EVCC_DC_VTB_SDP_004'.

TC Id	TC_EVCC_DC_VTB_SDP_004
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5 % duty cycle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 99 — Test case description for 'TC_EVCC_DC_VTB_SDP_005'

Table 99 lists the test case description for 'TC_EVCC_DC_VTB_SDP_005'.

TC Id	TC_EVCC_DC_VTB_SDP_005
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 100 — Test case description for 'TC_EVCC_DC_VTB_SDP_006'

Table 100 lists the test case description for 'TC_EVCC_DC_VTB_SDP_006'.

TC Id	TC_EVCC_DC_VTB_SDP_006
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]

Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 101 — Test case description for 'TC_EVCC_DC_VTB_SDP_007'

Table 101 lists the test case description for 'TC_EVCC_DC_VTB_SDP_007'.

TC Id	TC_EVCC_DC_VTB_SDP_007
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 102 — Test case description for 'TC_EVCC_DC_VTB_SDP_008'

Table 102 lists the test case description for 'TC_EVCC_DC_VTB_SDP_008'.

TC Id	TC_EVCC_DC_VTB_SDP_008
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication

	session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 103 — Test case description for 'TC_EVCC_DC_VTB_SDP_009'

Table 103 lists the test case description for 'TC_EVCC_DC_VTB_SDP_009'.

TC Id	TC_EVCC_DC_VTB_SDP_009
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element=0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002

PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 104 — Test case description for 'TC_EVCC_DC_VTB_SDP_010'

Table 104 lists the test case description for 'TC_EVCC_DC_VTB_SDP_010'.

TC Id	TC_EVCC_DC_VTB_SDP_010
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element=0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 105 — Test case description for 'TC_EVCC_DC_VTB_SDP_011'

Table 105 lists the test case description for 'TC_EVCC_DC_VTB_SDP_011'.

TC Id	TC_EVCC_DC_VTB_SDP_011
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element=0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 106 — Test case description for 'TC_EVCC_DC_VTB_SDP_012'

Table 106 lists the test case description for 'TC_EVCC_DC_VTB_SDP_012'.

TC Id	TC_EVCC_DC_VTB_SDP_012
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element=0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS

	Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 107 — Test case description for 'TC_EVCC_DC_VTB_SDP_013'

Table 107 lists the test case description for 'TC_EVCC_DC_VTB_SDP_013'.

TC Id	TC_EVCC_DC_VTB_SDP_013
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element=0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	

f_EVCC_CMN_TB_VTB_SDP_002

Table 108 — Test case description for 'TC_EVCC_DC_VTB_SDP_014'

Table 108 lists the test case description for 'TC_EVCC_DC_VTB_SDP_014'.

TC Id	TC_EVCC_DC_VTB_SDP_014
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element=0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 109 — Test case description for 'TC_EVCC_DC_VTB_SDP_015'

Table 109 lists the test case description for 'TC_EVCC_DC_VTB_SDP_015'.

TC Id	TC_EVCC_DC_VTB_SDP_015
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element=0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an

	SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 110 — Test case description for 'TC_EVCC_DC_VTB_SDP_016'

Table 110 lists the test case description for 'TC_EVCC_DC_VTB_SDP_016'.

TC Id	TC_EVCC_DC_VTB_SDP_016
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element=0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.10.1.4; 15118-2:7.10.1.6; 15118-2:7.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	

PreCondition
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001
Expected behavior
f_EVCC_CMN_TB_VTB_SDP_002

10 Test case descriptions for 15118-2 V2G application layer messages

10.1 General information

This clause covers all test case descriptions for the *V2G Application layer message requirements* as defined in ISO 15118-2.

10.2 SECC test cases

This subclause covers all V2G Application layer test case descriptions, where the *SECC* is defined as SUT.

10.2.1 V2G protocol handshake

This subclause covers the *V2G protocol handshake* test case descriptions, where the *SECC* is defined as SUT.

10.2.1.1 SECC test cases for SupportedAppProtocol

Table 111 lists the test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_001'.

Table 111 — Test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_001'

TC Id	TC_SECC_CMN_VTB_SupportedAppProtocol_001
Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolReq message with a list of valid AppProtocols including ISO namespace and all additional mandatory parameters. Test System then checks that the SUT sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation' or 'OK_SuccessfulNegotiationWithMinorDeviation' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.2.1; 15118-2:8.2.2; 15118-2:8.2.3; 15118-2:8.8.4.3.1; 15118-2:7.4; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.7.3.4; 15118-2:7.10.1.7; 15118-2:7.7.3.3; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-165], [V2G2-168], [V2G2-169], [V2G2-176], [V2G2-178], [V2G2-541], [V2G2-026], [V2G2-030], [V2G2-031], [V2G2-033], [V2G2-377], [V2G2-381], [V2G2-391], [V2G2-395], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-721], [V2G2-602], [V2G2-626], [V2G2-665], [V2G2-068], [V2G2-871], [V2G2-852]_eIM
Config Id	CF_04_001
PICS selection	
PIXIT selection	PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_Binding_001	
Expected behavior	
f_SECC_CMN_PR_InitConfiguration_002, f_SECC_CMN_TB_VTB_SupportedAppProtocol_001	

Table 112 lists the test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_002'.

Table 112 — Test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_002'

TC Id	TC_SECC_CMN_VTB_SupportedAppProtocol_002
Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolReq message with an invalid ISO namespace and all additional mandatory parameters. Test System then checks that the SUT sends a SupportedAppProtocolRes message with ResponseCode 'FAILED_NoNegotiation' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.2.1; 15118-2.8.2.2; 15118-2.8.2.3; 15118-2.8.8.3.1; 15118-2.7.4; 15118-2.7.7.3.3
Referenced requirement(s)	[V2G2-165], [V2G2-168], [V2G2-172], [V2G2-176], [V2G2-178], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-026], [V2G2-030], [V2G2-031], [V2G2-033], [V2G2-721], [V2G2-068]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Binding_001	
Expected behavior	
f_SECC_CMN_PR_InitConfiguration_002, f_SECC_CMN_TB_VTB_SupportedAppProtocol_002	

Table 113 lists the test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_003'.

Table 113 — Test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_003'

TC Id	TC_SECC_CMN_VTB_SupportedAppProtocol_003
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a SDP response message with valid security option, 'transportProtocol' equals to '0x00'H and a valid port and IP address. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.8.4.3.1; 15118-2.8.7.2.1; 15118-2.8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-536], [V2G2-537], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Binding_001	
Expected behavior	
f_SECC_CMN_PR_InitConfiguration_002, f_SECC_CMN_TB_VTB_SupportedAppProtocol_003	

Table 114 lists the test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_004'.

Table 114 — Test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_004'

TC Id	TC_SECC_CMN_VTB_SupportedAppProtocol_004
Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolReq message with a list of valid AppProtocols including ISO namespace and all additional mandatory parameters. Test System then checks that the SUT sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation' or 'OK_SuccessfulNegotiationWithMinorDeviation' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.2.1; 15118-2:8.2.2; 15118-2:8.2.3; 15118-2:8.8.4.3.1; 15118-2:7.4; 15118-2:8.6.2.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.7.3.3; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-165], [V2G2-168], [V2G2-169], [V2G2-176], [V2G2-178], [V2G2-541], [V2G2-026], [V2G2-030], [V2G2-031], [V2G2-033], [V2G2-377], [V2G2-391], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-721], [V2G2-665], [V2G2-068], [V2G2-852]_eIM
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_TLS := false
PreCondition	
f_SECC_CMN_PR_Binding_001	
Expected behavior	
f_SECC_CMN_PR_InitConfiguration_002, f_SECC_CMN_TB_VTB_SupportedAppProtocol_001	

Table 115 lists the test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_005'.

Table 115 — Test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_005'

TC Id	TC_SECC_CMN_VTB_SupportedAppProtocol_005
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_SECC_CommunicationSetup_Timeout' timer has expired on SUT side Test System then checks that the SUT stops waiting for establishing the TCP/TLS connection and does not respond to a TCP/TLS handshake request.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.7.3.1
Referenced requirement(s)	[V2G2-032], [V2G2-605]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Binding_001	
Expected behavior	
f_SECC_CMN_PR_InitConfiguration_002	

Table 116 lists the test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_006'.

Table 116 — Test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_006'

TC Id	TC_SECC_CMN_VTB_SupportedAppProtocol_006

Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolReq message with a list of valid AppProtocols (VersionNumberMinor does not match VersionNumberMinor of the SUT) including ISO namespace and all additional mandatory parameters. Test System then checks that the SUT sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiationWithMinorDeviation' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.2.1; 15118-2:8.2.2; 15118-2:8.2.3; 15118-2:8.8.4.3.1; 15118-2:7.4; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.7.3.4; 15118-2:7.10.1.7; 15118-2:7.7.3.3; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-165], [V2G2-168], [V2G2-169], [V2G2-176], [V2G2-178], [V2G2-541], [V2G2-026], [V2G2-030], [V2G2-031], [V2G2-033], [V2G2-377], [V2G2-381], [V2G2-391], [V2G2-395], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-721], [V2G2-602], [V2G2-626], [V2G2-665], [V2G2-068], [V2G2-871], [V2G2-170], [V2G2-852]_eIM
Config Id	CF_04_001
PICS selection	
PIXIT selection	PIXIT_SECC_CMN_VersionNumberMinor
PreCondition	
f_SECC_CMN_PR_Binding_001	
Expected behavior	
f_SECC_CMN_PR_InitConfiguration_002, f_SECC_CMN_TB_VTB_SupportedAppProtocol_004	

10.2.2 V2G messages

This subclause includes all *V2G message* test case descriptions, where the SECC is defined as SUT.

10.2.2.1 SECC test cases for SessionSetup

The following subclauses include all *V2G SessionSetup message* test case descriptions, where the SECC is defined as SUT.

10.2.2.1.1 Common test cases

Table 117 lists the test case description for 'TC_SECC_CMN_VTB_SessionSetup_001'.

Table 117 — Test case description for 'TC_SECC_CMN_VTB_SessionSetup_001'

TC Id	TC_SECC_CMN_VTB_SessionSetup_001
Test objective	Test System executes GoodCase procedure and sends a SessionSetupReq message with SessionID '0', valid EvccID and all additional mandatory parameters. Test System then checks that the SUT sends a SessionSetupRes message with a new SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-750], [V2G2-377], [V2G2-381], [V2G2-391], [V2G2-395], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-

	192], [V2G2-665], [V2G2-852]_eIM
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_001	

Table 118 lists the test case description for 'TC_SECC_CMN_VTB_SessionSetup_002'.

Table 118 — Test case description for 'TC_SECC_CMN_VTB_SessionSetup_002'

TC Id	TC_SECC_CMN_VTB_SessionSetup_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a SessionSetupReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.4.3.3.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_002	

Table 119 lists the test case description for 'TC_SECC_CMN_VTB_SessionSetup_003'.

Table 119 — Test case description for 'TC_SECC_CMN_VTB_SessionSetup_003'

TC Id	TC_SECC_CMN_VTB_SessionSetup_003
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a SupportedAppProtocolResmessage with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001

PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_003	

Table 120 lists the test case description for 'TC_SECC_CMN_VTB_SessionSetup_004'.

Table 120 — Test case description for 'TC_SECC_CMN_VTB_SessionSetup_004'

TC Id	TC_SECC_CMN_VTB_SessionSetup_004
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_SECC_CommunicationSetup_Timeout' timer has expired after receiving a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.3.1; 15118-2:8.7.3.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-714], [V2G2-715], [V2G2-716], [V2G2-605]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_004	

10.2.2.1.2 AC specific test cases

Table 121 lists the test case description for 'TC_SECC_AC_VTB_SessionSetup_001'.

Table 121 — Test case description for 'TC_SECC_AC_VTB_SessionSetup_001'

TC Id	TC_SECC_AC_VTB_SessionSetup_001
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SessionSetupReq message with the previous sessionID, valid EvccID and all additional mandatory parameters. Test System then checks that the SUT sends a SessionSetupRes message with the previous SessionID, ResponseCode 'OK_OldSessionJoined', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-

	2:8.4.3.2.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-377], [V2G2-381], [V2G2-463], [V2G2-741], [V2G2-745], [V2G2-753], [V2G2-754], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_005	

Table 122 lists the test case description for 'TC_SECC_AC_VTB_SessionSetup_002'.

Table 122 — Test case description for 'TC_SECC_AC_VTB_SessionSetup_002'

TC Id	TC_SECC_AC_VTB_SessionSetup_002
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SessionSetupReq message the with a new SessionID, valid EvccID and all additional mandatory parameters. Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-377], [V2G2-381], [V2G2-756], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_006	

Table 123 lists the test case description for 'TC_SECC_AC_VTB_SessionSetup_003'.

Table 123 — Test case description for 'TC_SECC_AC_VTB_SessionSetup_003'

TC Id	TC_SECC_AC_VTB_SessionSetup_003
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SessionSetupReq message with a SessionID '0', valid EvccID and all additional mandatory parameters. Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-377], [V2G2-381], [V2G2-750], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_006	

Table 124 lists the test case description for 'TC_SECC_AC_VTB_SessionSetup_004'.

Table 124 — Test case description for 'TC_SECC_AC_VTB_SessionSetup_004'

TC Id	TC_SECC_AC_VTB_SessionSetup_004
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SessionSetupReq message with the previous sessionID, valid EvccID and all additional mandatory parameters. Test System then checks that the SUT sends a SessionSetupRes message with the previous SessionID, ResponseCode 'OK_OldSessionJoined', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS

	Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-377], [V2G2-381], [V2G2-463], [V2G2-741], [V2G2-745], [V2G2-753], [V2G2-754], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_005	

Table 125 lists the test case description for 'TC_SECC_AC_VTB_SessionSetup_005'.

Table 125 — Test case description for 'TC_SECC_AC_VTB_SessionSetup_005'

TC Id	TC_SECC_AC_VTB_SessionSetup_005
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SessionSetupReq message the with a new SessionID, valid EvccID and all additional mandatory parameters. Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid Evseld and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-377], [V2G2-381], [V2G2-756], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_006	

Table 126 lists the test case description for 'TC_SECC_AC_VTB_SessionSetup_006'.

Table 126 — Test case description for 'TC_SECC_AC_VTB_SessionSetup_006'

TC Id	TC_SECC_AC_VTB_SessionSetup_006
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SessionSetupReq message with a SessionID '0', valid EvccID and all additional mandatory parameters. Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-377], [V2G2-381], [V2G2-750], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_006	

10.2.2.1.3 DC specific test cases

Table 127 lists the test case description for 'TC_SECC_DC_VTB_SessionSetup_001'.

Table 127 — Test case description for 'TC_SECC_DC_VTB_SessionSetup_001'

TC Id	TC_SECC_DC_VTB_SessionSetup_001
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SessionSetupReq message with the previous sessionID, valid EvccID and all additional mandatory parameters. Test System then checks that the SUT sends a SessionSetupRes message with the previous SessionID, ResponseCode 'OK_OldSessionJoined', valid EvseId and all

	additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-391], [V2G2-395], [V2G2-463], [V2G2-741], [V2G2-745], [V2G2-753], [V2G2-754], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_005	

Table 128 lists the test case description for 'TC_SECC_DC_VTB_SessionSetup_002'.

Table 128 — Test case description for 'TC_SECC_DC_VTB_SessionSetup_002'

TC Id	TC_SECC_DC_VTB_SessionSetup_002
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SessionSetupReq message the with a new SessionID, valid EvccID and all additional mandatory parameters. Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-391], [V2G2-395], [V2G2-756], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	

Expected behavior
f_SECC_CMN_TB_VTB_SessionSetup_006

Table 129 lists the test case description for 'TC_SECC_DC_VTB_SessionSetup_003'.

Table 129 — Test case description for 'TC_SECC_DC_VTB_SessionSetup_003'

TC Id	TC_SECC_DC_VTB_SessionSetup_003
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SessionSetupReq message with a SessionID '0', valid EvccID and all additional mandatory parameters. Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-391], [V2G2-395], [V2G2-750], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_006	

Table 130 lists the test case description for 'TC_SECC_DC_VTB_SessionSetup_004'.

Table 130 — Test case description for 'TC_SECC_DC_VTB_SessionSetup_004'

TC Id	TC_SECC_DC_VTB_SessionSetup_004
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SessionSetupReq message with the previous sessionID, valid EvccID and all additional mandatory parameters. Test System then checks that the SUT sends a SessionSetupRes message with the

	previous SessionID, ResponseCode 'OK_OldSessionJoined', valid Evseld and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-391], [V2G2-395], [V2G2-463], [V2G2-741], [V2G2-745], [V2G2-753], [V2G2-754], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_005	

Table 131 lists the test case description for 'TC_SECC_DC_VTB_SessionSetup_005'.

Table 131 — Test case description for 'TC_SECC_DC_VTB_SessionSetup_005'

TC Id	TC_SECC_DC_VTB_SessionSetup_005
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SessionSetupReq message the with a new SessionID, valid EvccID and all additional mandatory parameters. Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid Evseld and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-391], [V2G2-395], [V2G2-756], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f SECC DC PR SessionStop 001, f SECC startSleepingPhase,	

f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001
Expected behavior
f_SECC_CMN_TB_VTB_SessionSetup_006

Table 132 lists the test case description for 'TC_SECC_DC_VTB_SessionSetup_006'.

Table 132 — Test case description for 'TC_SECC_DC_VTB_SessionSetup_006'

TC Id	TC_SECC_DC_VTB_SessionSetup_006
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SessionSetupReq message with a SessionID '0', valid EvccID and all additional mandatory parameters. Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-391], [V2G2-395], [V2G2-750], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_006	

10.2.2.2 SECC test cases for ServiceDiscovery

The following subclauses include all *V2G ServiceDiscovery message* test case descriptions, where the SECC is defined as SUT.

10.2.2.2.1 Common test cases

Table 133 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_001'.

Table 133 — Test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_001'

TC Id	TC_SECC_CMN_VTB_ServiceDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the

	current SessionID, ResponseCode 'Ok', PaymentOption, valid SupportedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.2; 15118-2:8.4.3.3; 15118-2:8.8.4.3.1; 15118-2:8.5.2.9; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.5.2.4; 15118-2:8.7.4.4; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-265], [V2G2-266], [V2G2-267], [V2G2-268], [V2G2-271], [V2G2-272], [V2G2-282], [V2G2-283], [V2G2-418], [V2G2-545], [V2G2-752], [V2G2-377], [V2G2-381], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-632], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-757], [V2G2-758], [V2G2-759], [V2G2-915], [V2G2-665], [V2G2-852]_eIM
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SessionSetup_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDiscovery_001	

Table 134 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_002'.

Table 134 — Test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_002'

TC Id	TC_SECC_CMN_VTB_ServiceDiscovery_002
Test objective	Test System executes GoodCase procedure and sends a valid AuthorizationReq message instead of a ServiceDiscoveryReq message. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.7.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1;
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SessionSetup_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDiscovery_002	

Table 135 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_003'.

Table 135 — Test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_003'

TC Id	TC_SECC_CMN_VTB_ServiceDiscovery_003
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryReq message

	with an invalid SessionID and all additional mandatory parameters. Test System then checks that the SUT sends an ServiceDiscoveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.8.4.2; 15118-2.8.4.3.3.2; 15118-2.8.4.3.3.3; 15118-2.8.8.3.1; 15118-2.8.8.4.3.1;
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SessionSetup_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDiscovery_003	

Table 136 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_004'.

Table 136 — Test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_004'

TC Id	TC_SECC_CMN_VTB_ServiceDiscovery_004
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a SessionSetupRes message with a new SessionID, ResponseCode 'OK_NewSessionEstablished', valid Evseld and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.8.4.3.1; 15118-2.8.7.2.1; 15118-2.8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SessionSetup_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDiscovery_004	

10.2.2.2.2 AC specific test cases

Table 137 lists the test case description for 'TC_SECC_AC_VTB_ServiceDiscovery_001'.

Table 137 — Test case description for 'TC_SECC_AC_VTB_ServiceDiscovery_001'

TC Id	TC_SECC_AC_VTB_ServiceDiscovery_001
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test

	System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ServiceDiscoveryReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', all additional mandatory parameters and the PaymentOption and ChargeService which was selected by the EVCC in the previous session.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.3.2; 15118-2:8.4.3.3.3; 15118-2:8.8.4.3.1; 15118-2:8.5.2.9; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.5.2.4; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-265], [V2G2-266], [V2G2-267], [V2G2-268], [V2G2-271], [V2G2-272], [V2G2-282], [V2G2-283], [V2G2-418], [V2G2-545], [V2G2-752], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-632], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-741], [V2G2-745], [V2G2-757], [V2G2-758], [V2G2-759], [V2G2-665], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDiscovery_005	

Table 138 lists the test case description for 'TC_SECC_AC_VTB_ServiceDiscovery_002'.

Table 138 — Test case description for 'TC_SECC_AC_VTB_ServiceDiscovery_002'

TC Id	TC_SECC_AC_VTB_ServiceDiscovery_002
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ServiceDiscoveryReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', all additional mandatory parameters and the PaymentOption and ChargeService which was selected by the EVCC in the previous session.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.3.2; 15118-2:8.4.3.3.3; 15118-2:8.8.4.3.1; 15118-2:8.5.2.9; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.5.2.4; 15118-2:8.7.4.2; 15118-2:7.4
Referenced	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-

requirement(s)	265], [V2G2-266], [V2G2-267], [V2G2-268], [V2G2-271], [V2G2-272], [V2G2-282], [V2G2-283], [V2G2-418], [V2G2-545], [V2G2-752], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-632], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-741], [V2G2-745], [V2G2-757], [V2G2-758], [V2G2-759], [V2G2-665], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDiscovery_005	

10.2.2.3 DC specific test cases

Table 139 lists the test case description for 'TC_SECC_DC_VTB_ServiceDiscovery_001'.

Table 139 — Test case description for 'TC_SECC_DC_VTB_ServiceDiscovery_001'

TC Id	TC_SECC_DC_VTB_ServiceDiscovery_001
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ServiceDiscoveryReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', all additional mandatory parameters and the PaymentOption and ChargeService which was selected by the EVCC in the previous session.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.3.2; 15118-2:8.4.3.3.3; 15118-2:8.4.3.1; 15118-2:8.5.2.9; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.5.2.4; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-265], [V2G2-266], [V2G2-267], [V2G2-268], [V2G2-271], [V2G2-272], [V2G2-282], [V2G2-283], [V2G2-418], [V2G2-545], [V2G2-752], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-632], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-741], [V2G2-745], [V2G2-757], [V2G2-758], [V2G2-759], [V2G2-665], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	

PreCondition
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SessionSetupPausedSession_001
Expected behavior
f_SECC_CMN_TB_VTB_ServiceDiscovery_005

Table 140 lists the test case description for 'TC_SECC_DC_VTB_ServiceDiscovery_002'.

Table 140 — Test case description for 'TC_SECC_DC_VTB_ServiceDiscovery_002'

TC Id	TC_SECC_DC_VTB_ServiceDiscovery_002
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ServiceDiscoveryReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', all additional mandatory parameters and the PaymentOption and ChargeService which was selected by the EVCC in the previous session.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.2; 15118-2:8.4.3.3; 15118-2:8.4.3.1; 15118-2:8.5.2.9; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.5.2.4; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-265], [V2G2-266], [V2G2-267], [V2G2-268], [V2G2-271], [V2G2-272], [V2G2-282], [V2G2-283], [V2G2-418], [V2G2-545], [V2G2-752], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-632], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-741], [V2G2-745], [V2G2-757], [V2G2-758], [V2G2-759], [V2G2-665], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
	f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SessionSetupPausedSession_001
Expected behavior	
	f_SECC_CMN_TB_VTB_ServiceDiscovery_005

10.2.2.3 SECC test cases for ServiceDetail

The following subclauses include all *V2G ServiceDetail* message test case descriptions, where the SECC is defined as SUT.

10.2.2.3.1 Common test cases

Table 141 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_001'.

Table 141 — Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_001'

TC Id	TC_SECC_CMN_VTB_ServiceDetail_001
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID, all additional mandatory parameters and the corresponding service information. If the SUT provides more than one service the Test System sends another valid ServiceDetailReq message and the SUT shall respond to this request message with the corresponding service information again until the required information for each service are received.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.4.2; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.4.1; 15118-2:8.6.3.5; 15118-2:8.6.3.6; 15118-2:8.5.2.21; 15118-2:8.5.2.22; 15118-2:8.5.2.23; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-199], [V2G2-200], [V2G2-548], [V2G2-379], [V2G2-393], [V2G2-414], [V2G2-416], [V2G2-419], [V2G2-420], [V2G2-421], [V2G2-426], [V2G2-427], [V2G2-343], [V2G2-344], [V2G2-345], [V2G2-346], [V2G2-347], [V2G2-348], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665], [V2G2-852]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_SECC_CMN_TLS := true, PIXIT_SECC_CMN_InternetAccess := unknown
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_001	

Table 142 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_002'.

Table 142 — Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_002'

TC Id	TC_SECC_CMN_VTB_ServiceDetail_002
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID, all additional mandatory parameters and the corresponding service information (certificate). If the SUT provides more than one service the Test System sends another valid ServiceDetailReq message and the SUT shall respond to this request message with the corresponding service information again until the required information for each service are received.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.4.2; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.5; 15118-2:8.6.3.6; 15118-2:8.5.2.21; 15118-2:8.5.2.22; 15118-2:8.5.2.23; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-199], [V2G2-200], [V2G2-548], [V2G2-385], [V2G2-397], [V2G2-414], [V2G2-416], [V2G2-419], [V2G2-420], [V2G2-421], [V2G2-426], [V2G2-427], [V2G2-428], [V2G2-343], [V2G2-344], [V2G2-345], [V2G2-346], [V2G2-347], [V2G2-348], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665]

Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_InternetAccess := unknown
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_001	

Table 143 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_003'.

Table 143 — Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_003'

TC Id	TC_SECC_CMN_VTB_ServiceDetail_003
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a ServiceDetailReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.2; 15118-2:8.4.3.3.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-195], [V2G2-196], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_002	

Table 144 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_004'.

Table 144 — Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_004'

TC Id	TC_SECC_CMN_VTB_ServiceDetail_004
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a ServiceDetailReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.2; 15118-2:8.4.3.3.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-195], [V2G2-196], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]

Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_002	

Table 145 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_005'.

Table 145 — Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_005'

TC Id	TC_SECC_CMN_VTB_ServiceDetail_005
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with an invalid SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2.7.4; 15118-2:8.4.2; 15118-2:8.4.3.4.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-199], [V2G2-200], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_003	

Table 146 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_006'.

Table 146 — Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_006'

TC Id	TC_SECC_CMN_VTB_ServiceDetail_006
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with an invalid SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2.7.4; 15118-2:8.4.2; 15118-2:8.4.3.4.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-199], [V2G2-200], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]

Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_003	

Table 147 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_007'.

Table 147 — Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_007'

TC Id	TC_SECC_CMN_VTB_ServiceDetail_007
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with the current SessionID, invalid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_ServiceIDInvalid' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.2; 15118-2:8.4.3.4.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1; 15118-2:8.6.3.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-199], [V2G2-200], [V2G2-464], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-425]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_004	

Table 148 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_008'.

Table 148 — Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_008'

TC Id	TC_SECC_CMN_VTB_ServiceDetail_008
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with the current SessionID, invalid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_ServiceIDInvalid' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.2; 15118-2:8.4.3.4.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1; 15118-2:8.6.3.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-199], [V2G2-200], [V2G2-464], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-425]
Config Id	CF_04_001

PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_004	

Table 149 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_009'.

Table 149 — Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_009'

TC Id	TC_SECC_CMN_VTB_ServiceDetail_009
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_005	

Table 150 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_010'.

Table 150 — Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_010'

TC Id	TC_SECC_CMN_VTB_ServiceDetail_010
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName

	:= PKI_1
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_005	

Table 151 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_011'.

Table 151 — Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_011'

TC Id	TC_SECC_CMN_VTB_ServiceDetail_011
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID, all additional mandatory parameters and the corresponding service information (internetAccess). If the SUT provides more than one service the Test System sends another valid ServiceDetailReq message and the SUT shall respond to this request message with the corresponding service information again until the required information for each service are received.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.4.2; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.4.1; 15118-2:8.6.3.5; 15118-2:8.6.3.6; 15118-2:8.5.2.21; 15118-2:8.5.2.22; 15118-2:8.5.2.23; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-199], [V2G2-200], [V2G2-548], [V2G2-379], [V2G2-393], [V2G2-414], [V2G2-416], [V2G2-419], [V2G2-420], [V2G2-421], [V2G2-426], [V2G2-427], [V2G2-343], [V2G2-344], [V2G2-345], [V2G2-346], [V2G2-347], [V2G2-348], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665], [V2G2-429], [V2G2-852]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_SECC_CMN_TLS := true, PIXIT_SECC_CMN_InternetAccess := internetAccess
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_001	

Table 152 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_012'.

Table 152 — Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_012'

TC Id	TC_SECC_CMN_VTB_ServiceDetail_012
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID, all additional mandatory parameters and the corresponding service information (internetAccess, certificate). If the SUT provides more than one service the Test System sends another valid ServiceDetailReq message and the SUT shall respond to this request message with the corresponding service information again until the required information for each

	service are received.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.4.2; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.5; 15118-2:8.6.3.6; 15118-2:8.5.2.21; 15118-2:8.5.2.22; 15118-2:8.5.2.23; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-199], [V2G2-200], [V2G2-548], [V2G2-385], [V2G2-397], [V2G2-414], [V2G2-416], [V2G2-419], [V2G2-420], [V2G2-421], [V2G2-426], [V2G2-427], [V2G2-428], [V2G2-343], [V2G2-344], [V2G2-345], [V2G2-346], [V2G2-347], [V2G2-348], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665], [V2G2-429]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_InternetAccess := internetAccess
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_001	

10.2.2.4 SECC test cases for PaymentServiceSelection

The following subclauses include all V2G *PaymentServiceSelection* message test case descriptions, where the SECC is defined as SUT.

10.2.2.4.1 Common test cases

Table 153 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_001'.

Table 153 — Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_001'

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_001
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-551], [V2G2-752], [V2G2-377], [V2G2-381], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-636], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665], [V2G2-852]_eIM
Config Id	CF_04_001
PICS selection	
PIXIT selection	PIXIT_SECC_CMN_VAS := none_
PreCondition	

f_SECC_CMN_PR_ServiceDiscovery_001
Expected behavior
f_SECC_CMN_TB_VTB_PaymentServiceSelection_001

Table 154 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_002'.

Table 154 — Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_002'

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_002
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'ExternalPayment', valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-551], [V2G2-752], [V2G2-377], [V2G2-391], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665], [V2G2-852]_eIM
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_001	

Table 155 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_003'.

Table 155 — Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_003'

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_003
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'Contract', valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-551], [V2G2-752], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName

	:= PKI_1
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_001	

Table 156 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_004'.

Table 156 — Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_004'

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_004
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a PaymentServiceSelectionReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.8.4.3.3.3; 15118-2.8.8.3.1; 15118-2.8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_003
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_002	

Table 157 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_005'.

Table 157 — Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_005'

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_005
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with an invalid SessionID, SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentServiceSelectionRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.8.4.3.5.3; 15118-2.8.8.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_003
PICS selection	

PIXIT selection	
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_003	

Table 158 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_006'.

Table 158 — Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_006'

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_006
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption, invalid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_ServiceSelectionInvalid' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.5.3; 15118-2:8.8.3.1; 15118-2:8.6.3.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-752], [V2G2-467], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-433]
Config Id	CF_04_003
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_004	

Table 159 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_007'.

Table 159 — Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_007'

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_007
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption, all additional mandatory parameters and without a ServiceID value equal to 1. Test System then checks that the SUT sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_NoChargeServiceSelected' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.5.3; 15118-2:8.8.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-752], [V2G2-804], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_003
PICS selection	

PIXIT selection	
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_005	

Table 160 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_008'.

Table 160 — Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_008'

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_008
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_003
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_006	

Table 161 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_009'.

Table 161 — Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_009'

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_009
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'contract', valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_PaymentSelectionInvalid' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.5.3; 15118-2:8.8.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-752], [V2G2-465], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_003
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_PaymentOption := externalPayment
PreCondition	

f_SECC_CMN_PR_ServiceDetail_001
Expected behavior
f_SECC_CMN_TB_VTB_PaymentServiceSelection_007

Table 162 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_010'.

Table 162 — Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_010'

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_010
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'externalPayment', valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_PaymentSelectionInvalid' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.5.3; 15118-2:8.8.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-752], [V2G2-465], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_003
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_PaymentOption := contract
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_007	

10.2.2.5 SECC test cases for PaymentDetails

The following subclauses include all V2G *PaymentDetails* message test case descriptions, where the SECC is defined as SUT.

10.2.2.5.1 Common test cases

Table 163 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_001'.

Table 163 — Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_001'

TC Id	TC_SECC_CMN_VTB_PaymentDetails_001
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.6.2; 15118-2:8.4.3.6.3; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-560], [V2G2-752], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665],

	[V2G2-666], [V2G2-667], [V2G2-915], [V2G2-899], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := _none, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_001	

Table 164 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_002'.

Table 164 — Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_002'

TC Id	TC_SECC_CMN_VTB_PaymentDetails_002
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.6.2; 15118-2:8.4.3.6.3; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-560], [V2G2-752], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-899], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_001	

Table 165 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_003'.

Table 165 — Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_003'

TC Id	TC_SECC_CMN_VTB_PaymentDetails_003
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.6.2; 15118-

	2:8.4.3.6.3; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-560], [V2G2-752], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-899], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_001	

Table 166 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_004'.

Table 166 — Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_004'

TC Id	TC_SECC_CMN_VTB_PaymentDetails_004
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a PaymentDetailsReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.3.3; 15118-2:8.4.3.6.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := _none, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_002	

Table 167 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_005'.

Table 167 — Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_005'

TC Id	TC_SECC_CMN_VTB_PaymentDetails_005
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with an invalid SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP

	connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.8.4.3.6.2; 15118-2.8.4.3.6.3; 15118-2.8.8.3.1;
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := _none, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_003	

Table 168 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_006'.

Table 168 — Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_006'

TC Id	TC_SECC_CMN_VTB_PaymentDetails_006
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.8.4.3.1; 15118-2.8.7.2.1; 15118-2.8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := _none, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_004	

Table 169 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_007'.

Table 169 — Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_007'

TC Id	TC_SECC_CMN_VTB_PaymentDetails_007
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with the current SessionID, valid EMAID, invalid contract certificate chain (Expired) and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'FAILED_CertificateExpired' and all additional mandatory parameters and terminates the V2G communication session by closing the

	TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.6.2; 15118-2:8.4.3.6.3; 15118-2:8.8.3.1;
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-474], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-899]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := _none, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_005	

Table 170 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_008'.

Table 170 — Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_008'

TC Id	TC_SECC_CMN_VTB_PaymentDetails_008
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with the current SessionID, valid EMAID, invalid contract certificate chain (Invalid content ->Country name was set) and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'FAILED', 'FAILED_NoCertificateAvailable' or 'FAILED_CertificateNotAllowedAtThisEVSE' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.6.2; 15118-2:8.4.3.6.3; 15118-2:8.8.3.1;
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-899]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := _none, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_005	

Table 171 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_009'.

Table 171 — Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_009'

TC Id	TC_SECC_CMN_VTB_PaymentDetails_009
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with the current SessionID, valid EMAID, invalid contract certificate chain (Expired

	issuer Sub CA 1) and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'FAILED', 'FAILED_NoCertificateAvailable' or 'FAILED_CertificateNotAllowedAtThisEVSE' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.6.2; 15118-2:8.4.3.6.3; 15118-2:8.8.3.1;
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-899]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := _none, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_005	

Table 172 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_010'.

Table 172 — Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_010'

TC Id	TC_SECC_CMN_VTB_PaymentDetails_010
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with the current SessionID, valid EMAID, invalid contract certificate chain (Expired issuer Sub CA 2) and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'FAILED', 'FAILED_NoCertificateAvailable' or 'FAILED_CertificateNotAllowedAtThisEVSE' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.6.2; 15118-2:8.4.3.6.3; 15118-2:8.8.3.1;
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-899]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := _none, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_005	

10.2.2.6 SECC test cases for Authorization

The following subclauses include all *V2G Authorization message* test case descriptions, where the SECC is defined as SUT.

10.2.2.6.1 Common test cases

Table 173 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_001'.

Table 173 — Test case description for 'TC_SECC_CMN_VTB_Authorization_001'

TC Id	TC_SECC_CMN_VTB_Authorization_001
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. If a valid AuthorizationRes message with EVSEProcessing 'Ongoing_WaitingForCustomerInteraction' was received before, the Test System sends another AuthorizationReq message. This message sequence will be repeated until the authorization has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.2; 15118-2:8.7.4.2; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.4.1; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-563], [V2G2-752], [V2G2-856], [V2G2-377], [V2G2-391], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_AuthorizationOngoing := unknown
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_001	

Table 174 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_002'.

Table 174 — Test case description for 'TC_SECC_CMN_VTB_Authorization_002'

TC Id	TC_SECC_CMN_VTB_Authorization_002
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with a valid signature, current SessionID, valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. If a valid AuthorizationRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another AuthorizationReq message. This message sequence will be repeated until the authorization has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.2; 15118-2:8.7.4.2; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.3.2; 15118-2:7.9.2.4.5
Referenced	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-

requirement(s)	563], [V2G2-752], [V2G2-856], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-005], [V2G2-007], [V2G2-653], [V2G2-901], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_002	

Table 175 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_003'.

Table 175 — Test case description for 'TC_SECC_CMN_VTB_Authorization_003'

TC Id	TC_SECC_CMN_VTB_Authorization_003
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of an AuthorizationReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.3.3; 15118-2:8.7.4.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_003	

Table 175 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_003'.

Table 176 — Test case description for 'TC_SECC_CMN_VTB_Authorization_004'

TC Id	TC_SECC_CMN_VTB_Authorization_004
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of an AuthorizationReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.3.3; 15118-2:8.7.4.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-

requirement(s)	752], [V2G2-459], [V2G2-736], [V2G2-539], [V2G2-729], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_003	

Table 177 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_005'.

Table 177 — Test case description for 'TC_SECC_CMN_VTB_Authorization_005'

TC Id	TC_SECC_CMN_VTB_Authorization_005
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with the an invalid SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDiscoveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.2; 15118-2:8.7.4.2; 15118-2:8.8.3.1 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_004	

Table 178 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_006'.

Table 178 — Test case description for 'TC_SECC_CMN_VTB_Authorization_006'

TC Id	TC_SECC_CMN_VTB_Authorization_006
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with a valid signature, invalid SessionID, valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDiscoveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.2; 15118-2:8.7.4.2; 15118-2:8.8.3.1 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]

Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_005	

Table 179 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_007'.

Table 179 — Test case description for 'TC_SECC_CMN_VTB_Authorization_007'

TC Id	TC_SECC_CMN_VTB_Authorization_007
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with a valid signature, current SessionID, invalid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_ChallengeInvalid' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.2; 15118-2:8.7.4.2; 15118-2:8.8.3.1 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-752], [V2G2-475], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-901]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_006	

Table 180 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_008'.

Table 180 — Test case description for 'TC_SECC_CMN_VTB_Authorization_008'

TC Id	TC_SECC_CMN_VTB_Authorization_008
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001

PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_007	

Table 181 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_009'.

Table 181 — Test case description for 'TC_SECC_CMN_VTB_Authorization_009'

TC Id	TC_SECC_CMN_VTB_Authorization_009
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PaymentDetailsRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_007	

Table 182 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_010'.

Table 182 — Test case description for 'TC_SECC_CMN_VTB_Authorization_010'

TC Id	TC_SECC_CMN_VTB_Authorization_010
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with an invalid signature (wrong private key used), current SessionID, valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.2; 15118-2:8.7.4.2; 15118-2:8.8.3.1 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-752], [V2G2-461], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-901]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1

PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_008	

Table 183 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_011'.

Table 183 — Test case description for 'TC_SECC_CMN_VTB_Authorization_011'

TC Id	TC_SECC_CMN_VTB_Authorization_011
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with an invalid signature (signed body changed after signature calculation), current SessionID, valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.2; 15118-2:8.7.4.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-752], [V2G2-461], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-901]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_009	

Table 184 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_012'.

Table 184 — Test case description for 'TC_SECC_CMN_VTB_Authorization_012'

TC Id	TC_SECC_CMN_VTB_Authorization_012
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with a valid signature, current SessionID, valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. If a valid AuthorizationRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another AuthorizationReq message. This message sequence will be repeated until the authorization has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.2; 15118-2:8.7.4.2; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.3.2; 15118-2:7.9.2.4.5; 15118-2:7.9.2.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-563], [V2G2-752], [V2G2-856], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-005], [V2G2-007], [V2G2-

	653], [V2G2-820], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_002	

Table 185 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_013'.

Table 185 — Test case description for 'TC_SECC_CMN_VTB_Authorization_013'

TC Id	TC_SECC_CMN_VTB_Authorization_013
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with a valid signature, current SessionID, valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. If a valid AuthorizationRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another AuthorizationReq message. This message sequence will be repeated until the authorization has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.2; 15118-2:8.7.4.2; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.3.2; 15118-2:7.9.2.4.5; 15118-2:7.9.2.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-563], [V2G2-752], [V2G2-856], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-005], [V2G2-007], [V2G2-653], [V2G2-821], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_002	

Table 186 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_014'.

Table 186 — Test case description for 'TC_SECC_CMN_VTB_Authorization_014'

TC Id	TC_SECC_CMN_VTB_Authorization_014
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Furthermore the repetition of the Authorization message

	sequence will be checked (EVSEProcessing := 'Ongoing_WaitingForCustomerInteraction') until the 'V2G_SECC_Ongoing_Timer' has expired on SUT side. Test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.2; 15118-2:8.7.4.2; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.4.1; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.7.3.5; 15118-2:7.4; 15118-2:8.7.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-563], [V2G2-752], [V2G2-856], [V2G2-377], [V2G2-391], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665], [V2G2-687], [V2G2-712], [V2G2-713], [V2G2-729], [V2G2-034], [V2G2-435], [V2G2-854]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM, PICS_SECC_CMN_EIMDone := v2gAuthorization
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_010	

Table 187 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_015'.

Table 187 — Test case description for 'TC_SECC_CMN_VTB_Authorization_015'

TC Id	TC_SECC_CMN_VTB_Authorization_015
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Furthermore the repetition of the Authorization message sequence will be checked (EVSEProcessing := 'Ongoing') until the 'V2G_SECC_Ongoing_Timer' has expired on SUT side. Test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.2; 15118-2:8.7.4.2; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.3.2; 15118-2:7.9.2.4.5; 15118-2:8.7.3.5; 15118-2:7.4; 15118-2:8.7.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-563], [V2G2-752], [V2G2-856], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-005], [V2G2-007], [V2G2-653], [V2G2-901], [V2G2-665], [V2G2-687], [V2G2-712], [V2G2-713], [V2G2-729], [V2G2-034], [V2G2-435], [V2G2-855]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_AuthorizationOngoing := ongoing

PreCondition
f_SECC_CMN_PR_PaymentDetails_001
Expected behavior
f_SECC_CMN_TB_VTB_Authorization_011

Table 188 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_016'.

Table 188 — Test case description for 'TC_SECC_CMN_VTB_Authorization_016'

TC Id	TC_SECC_CMN_VTB_Authorization_016
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Furthermore the repetition of the Authorization message sequence will be checked (EVSEProcessing := 'Ongoing') until the 'V2G_SECC_Ongoing_Timer' has expired on SUT side. Test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.2; 15118-2:8.7.4.2; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.3.2; 15118-2:7.9.2.4.5; 15118-2:7.9.2.4.3; 15118-2:8.7.3.5; 15118-2:7.4; 15118-2:8.7.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-563], [V2G2-752], [V2G2-856], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-005], [V2G2-007], [V2G2-653], [V2G2-820], [V2G2-665], [V2G2-687], [V2G2-712], [V2G2-713], [V2G2-729], [V2G2-034], [V2G2-435], [V2G2-855]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_AuthorizationOngoing := ongoing
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_011	

Table 189 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_017'.

Table 189 — Test case description for 'TC_SECC_CMN_VTB_Authorization_017'

TC Id	TC_SECC_CMN_VTB_Authorization_017
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Furthermore the repetition of the Authorization message sequence will be checked (EVSEProcessing := 'Ongoing') until the 'V2G_SECC_Ongoing_Timer' has expired on SUT side. Test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.2; 15118-2:8.7.4.2; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.3.2; 15118-2:7.9.2.4.5; 15118-2:7.9.2.4.3; 15118-2:8.7.3.5; 15118-2:7.4; 15118-2:8.7.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-563], [V2G2-752], [V2G2-856], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-005], [V2G2-007], [V2G2-653], [V2G2-821], [V2G2-665], [V2G2-687], [V2G2-712], [V2G2-713], [V2G2-729], [V2G2-034], [V2G2-435], [V2G2-855]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_AuthorizationOngoing := ongoing
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_011	

10.2.2.7 SECC test cases for ChargeParameterDiscovery

The following subclauses include all *V2G ChargeParameterDiscovery message* test case descriptions, where the SECC is defined as SUT.

10.2.2.7.1 Common test cases

Table 190 lists the test case description for 'TC_SECC_CMN_VTB_ChargeParameterDiscovery_001'.

Table 190 — Test case description for 'TC_SECC_CMN_VTB_ChargeParameterDiscovery_001'

TC Id	TC_SECC_CMN_VTB_ChargeParameterDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a ChargeParameterDiscoveryReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.3.3; 15118-2:8.4.3.8.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1; 15118-2:8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ChargeParameterDiscovery_001	

Table 191 lists the test case description for 'TC_SECC_CMN_VTB_ChargeParameterDiscovery_002'.

Table 191 — Test case description for 'TC_SECC_CMN_VTB_ChargeParameterDiscovery_002'

TC Id	TC_SECC_CMN_VTB_ChargeParameterDiscovery_002
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ChargeParameterDiscovery_002	

10.2.2.7.2 AC specific test cases

Table 192 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_001'.

Table 192 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_001'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665]

Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 193 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_002'.

Table 193 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_002'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_002
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 194 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_003'.

Table 194 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_003'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_003

Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with an invalid SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_002	

Table 195 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_004'.

Table 195 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_004'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_004
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, an invalid RequestedEnergyTransferType 'dC_extended' and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongEnergyTransferMode' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-752], [V2G2-476], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_003	

Table 196 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_005'.

Table 196 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_005'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_005

Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, valid RequestedEnergyTransferType, EVMaxCurrent '-100' and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongChargeParameter' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.8.4.2; 15118-2.8.4.3.8.3; 15118-2.8.8.3.1; 15118-2.8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-752], [V2G2-477], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_004	

Table 197 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_006'.

Table 197 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_006'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_006
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.8.4.2; 15118-2.8.4.3.8.3; 15118-2.8.8.4.3.2; 15118-2.8.5.2.7; 15118-2.8.5.2.12; 15118-2.8.5.2.13; 15118-2.8.5.2.14; 15118-2.8.5.2.15; 15118-2.8.5.2.18; 15118-2.8.5.3.1; 15118-2.8.5.3.3; 15118-2.8.6.2.3.1; 15118-2.8.6.2.3.2; 15118-2.8.8.3.1; 15118-2.7.3.4; 15118-2.8.6.2.1; 15118-2.8.8.4.1; 15118-2.8.4.3.8.3; 15118-2.8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown

PreCondition
f_SECC_CMN_PR_Authorization_001
Expected behavior
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001

Table 198 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_007'.

Table 198 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_007'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_007
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
	f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001
Expected behavior	
	f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005

Table 199 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_008'.

Table 199 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_008'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_008
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.3; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 200 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_009'.

Table 200 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_009'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_009
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has

	finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.5.2.16; 15118-2:8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 201 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_010'.

Table 201 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_010'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_010
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.3; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-

	666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 202 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_011'.

Table 202 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_011'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_011
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true,

	PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007	

Table 203 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_012'.

Table 203 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_012'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_012
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 204 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_013'.

Table 204 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_013'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_013
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a

	MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.3; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 205 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_014'.

Table 205 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_014'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_014
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Furthermore the repetition of the ChargeParameterDiscovery message sequence will be checked (EVSEProcessing := 'Ongoing') until the 'V2G_SECC_Ongoing_Timer' has expired on SUT side. Test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-

	2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.3; 15118-2:8.7.3.5; 15118-2:7.4; 15118-2:8.7.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665], [V2G2-688], [V2G2-712], [V2G2-713], [V2G2-729], [V2G2-034], [V2G2-435]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing := ongoing
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_008	

Table 206 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_015'.

Table 206 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_015'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_015
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.3; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true

PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 207 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_016'.

Table 207 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_016'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_016
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 208 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_017'.

Table 208 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_017'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_017
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network

	by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 209 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_018'.

Table 209 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_018'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_018
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without

	MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	<pre>PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false</pre>
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007	

Table 210 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_019'.

Table 210 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_019'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_019
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid

	scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 211 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_020'.

Table 211 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_020'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_020
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until

	the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007	

Table 212 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_021'.

Table 212 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_021'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_021
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007	

Table 213 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_022'.

Table 213 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_022'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_022
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13;

	15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 214 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_023'.

Table 214 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_023'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_023
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-303]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false

PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 215 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_024'.

Table 215 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_024'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_024
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.5.2.16; 15118-2:8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 216 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_025'.

Table 216 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_025'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_025

Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 217 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_026'.

Table 217 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_026'

TCId	TC_SECC_AC_VTB_ChargeParameterDiscovery_026
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13;

	15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.5.2.16; 15118-2:8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 218 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_027'.

Table 218 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_027'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_027
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.5.2.16; 15118-2:8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303]
Config Id	CF_04_001

PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 219 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_028'.

Table 219 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_028'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_028
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-303]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 220 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_029'.

Table 220 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_029'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_029
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 221 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_030'.

Table 221 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_030'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_030
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a

	valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007	

Table 222 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_031'.

Table 222 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_031'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_031
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then

	executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 223 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_032'.

Table 223 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_032'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_032
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid

	scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007	

Table 224 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_033'.

Table 224 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_033'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_033
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another

	ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 225 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_034'.

Table 225 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_034'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_034
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS

	Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007	

Table 226 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_035'.

Table 226 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_035'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_035
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-

	2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007	

Table 227 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_036'.

Table 227 — Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_036'

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_036
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.3.1; 15118-2:8.5.3.3; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:7.4

Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005	

10.2.2.7.3 DC specific test cases

Table 228 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_001'.

Table 228 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_001'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown

PreCondition
f_SECC_CMN_PR_Authorization_001
Expected behavior
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001

Table 229 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_002'.

Table 229 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_002'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_002
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with an invalid SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dc
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_002	

Table 230 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_003'.

Table 230 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_003'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_003
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, an invalid RequestedEnergyTransferType 'dC_extended' and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongEnergyTransferMode' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-752], [V2G2-476], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	

f_SECC_CMN_PR_Authorization_001
Expected behavior
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_003

Table 231 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_004'.

Table 231 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_004'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_004
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, valid RequestedEnergyTransferType, EVMaxCurrent '-100' and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongChargeParameter' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-752], [V2G2-477], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_004	

Table 232 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_005'.

Table 232 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_005'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_005
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667],

	[V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 233 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_006'.

Table 233 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_006'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_006
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 234 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_007'.

Table 234 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_007'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_007
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 235 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_008'.

Table 235 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_008'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_008

Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 236 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_009'.

Table 236 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_009'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_009
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4;

	15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.4; 15118-2:8.5.2.16; 15118-2:8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 237 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_010'.

Table 237 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_010'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_010
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime

	:= true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 238 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_011'.

Table 238 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_011'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_011
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	

f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001
Expected behavior
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007

Table 239 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_012'.

Table 239 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_012'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_012
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 240 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_013'.

Table 240 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_013'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_013
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid

	scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dc, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 241 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_014'.

Table 241 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_014'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_014
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Furthermore the repetition of the ChargeParameterDiscovery message sequence will be checked (EVSEProcessing := 'Ongoing') until the 'V2G_SECC_Ongoing_Timer' has expired on SUT side. Test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.3.5; 15118-2:7.4; 15118-2:8.7.2.1
Referenced	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299],

requirement(s)	[V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-688], [V2G2-712], [V2G2-713], [V2G2-729], [V2G2-034], [V2G2-435]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing := ongoing
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_008	

Table 242 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_015'.

Table 242 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_015'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_015
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	

Expected behavior
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006

Table 243 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_016'.

Table 243 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_016'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_016
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 244 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_017'.

Table 244 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_017'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_017
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the

	current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 245 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_018'.

Table 245 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_018'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_018
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message

	with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-452], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dc, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007	

Table 246 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_019'.

Table 246 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_019'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_019
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous

	session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 247 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_020'.

Table 247 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_020'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_020
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until

	the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007	

Table 248 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_021'.

Table 248 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_021'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_021
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007	

Table 249 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_022'.

Table 249 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_022'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_022
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-

	2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 250 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_023'.

Table 250 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_023'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_023
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-303]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true,

	PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 251 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_024'.

Table 251 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_024'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_024
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.4; 15118-2:8.5.2.16; 15118-2:8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 252 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_025'.

Table 252 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_025'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_025
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 253 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_026'.

Table 253 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_026'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_026
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS

	Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.4; 15118-2:8.5.2.16; 15118-2:8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 254 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_027'.

Table 254 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_027'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_027
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.4; 15118-2:8.5.2.16; 15118-2:8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303]
Config Id	CF_04_001

PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 255 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_028'.

Table 255 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_028'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_028
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-303]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 256 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_029'.

Table 256 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_029'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_029
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 257 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_030'.

Table 257 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_030'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_030

Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	<pre>PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false</pre>
PIXIT selection	<pre>PIXIT_SECC_CMN_SalesTariff := salesTariff</pre>
PreCondition	
<pre>f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001</pre>	
Expected behavior	
<pre>f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007</pre>	

Table 258 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_031'.

Table 258 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_031'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_031
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a

	valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 259 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_032'.

Table 259 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_032'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_032
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link

	detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007	

Table 260 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_033'.

Table 260 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_033'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_033
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a

	MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 261 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_034'.

Table 261 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_034'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_034
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message

	with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.3; 15118-2:8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007	

Table 262 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_035'.

Table 262 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_035'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_035
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional

	mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:8.5.2.16; 15118-2:8.5.2.17; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007	

Table 263 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_036'.

Table 263 — Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_036'

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_036
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT. Furthermore Test System checks that the SUT turns off the oscillator for the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing

	'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.2.12; 15118-2:8.5.2.13; 15118-2:8.5.2.14; 15118-2:8.5.2.15; 15118-2:8.5.2.18; 15118-2:8.5.4.1; 15118-2:8.5.4.4; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.4.3.8.3; 15118-2:8.7.4.4; 15118-2:8.7.4.2; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

10.2.2.8 SECC test cases for PowerDelivery

The following subclauses include all *V2G PowerDelivery message* test case descriptions, where the SECC is defined as SUT.

10.2.2.8.1 AC specific test cases

Table 264 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_001'.

Table 264 — Test case description for 'TC_SECC_AC_VTB_PowerDelivery_001'

TC Id	TC_SECC_AC_VTB_PowerDelivery_001
Test objective	Test System executes GoodCase procedure, sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', valid charging profile and TupleID and all additional mandatory parameters and signals CP State C. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.9.3; 15118-2:8.7.4.3; 15118-2:8.8.4.3.2; 15118-2:8.5.3.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-576], [V2G2-752], [V2G2-861], [V2G2-224], [V2G2-358], [V2G2-359], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-

	666], [V2G2-667], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_EvStateInChargeProgress := C
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_001	

Table 265 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_002'.

Table 265 — Test case description for 'TC_SECC_AC_VTB_PowerDelivery_002'

TC Id	TC_SECC_AC_VTB_PowerDelivery_002
Test objective	Test System executes GoodCase procedure, sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_' and all additional mandatory parameters and signals CP State D. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.9.3; 15118-2:8.7.4.3; 15118-2:8.8.4.3.2; 15118-2:8.5.3.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-576], [V2G2-752], [V2G2-861], [V2G2-224], [V2G2-358], [V2G2-359], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_EvStateInChargeProgress := D
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_001	

Table 266 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_003'.

Table 266 — Test case description for 'TC_SECC_AC_VTB_PowerDelivery_003'

TC Id	TC_SECC_AC_VTB_PowerDelivery_003
Test objective	Test System executes GoodCase procedure and initiates a renegotiation by sending a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.5.3.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1
Referenced	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-

requirement(s)	226], [V2G2-752], [V2G2-813], [V2G2-861], [V2G2-358], [V2G2-359], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown
PreCondition	
f_SECC_AC_PR_ChargingStatusOrMeteringReceiptRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_001	

Table 267 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_004'.

Table 267 — Test case description for 'TC_SECC_AC_VTB_PowerDelivery_004'

TC Id	TC_SECC_AC_VTB_PowerDelivery_004
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_' and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.7.4.3; 15118-2:8.8.4.3.2; 15118-2:8.5.3.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-576], [V2G2-752], [V2G2-861], [V2G2-358], [V2G2-359], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_001	

Table 268 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_005'.

Table 268 — Test case description for 'TC_SECC_AC_VTB_PowerDelivery_005'

TC Id	TC_SECC_AC_VTB_PowerDelivery_005
Test objective	Test System executes GoodCase procedure, sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters and signals CP State B. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS

	Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.7.4.3; 15118-2:8.8.4.3.1; 15118-2:8.5.3.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-568], [V2G2-752], [V2G2-864], [V2G2-358], [V2G2-359], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_001	

Table 269 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_006'.

Table 269 — Test case description for 'TC_SECC_AC_VTB_PowerDelivery_006'

TC Id	TC_SECC_AC_VTB_PowerDelivery_006
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a PowerDeliveryReq message with chargeProgressType 'Start'. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.3.3; 15118-2:8.7.4.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.2; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_002	

Table 270 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_007'.

Table 270 — Test case description for 'TC_SECC_AC_VTB_PowerDelivery_007'

TC Id	TC_SECC_AC_VTB_PowerDelivery_007
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a PowerDeliveryReq message with chargeProgressType 'Stop'. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the

	TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.3.3; 15118-2:8.7.4.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.2; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_002	

Table 271 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_008'.

Table 271 — Test case description for 'TC_SECC_AC_VTB_PowerDelivery_008'

TC Id	TC_SECC_AC_VTB_PowerDelivery_008
Test objective	Test System executes GoodCase procedure, sends a PowerDeliveryReq message with an invalid SessionID, ChargeProgress 'Start_', valid charging profile and TupleID and all additional mandatory parameters and signals CP State C or D. Test System then checks that the SUT sends a PowerDeliveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.3.3; 15118-2:8.7.4.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_002	

Table 272 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_009'.

Table 272 — Test case description for 'TC_SECC_AC_VTB_PowerDelivery_009'

TC Id	TC_SECC_AC_VTB_PowerDelivery_009
Test objective	Test System executes GoodCase procedure, sends a PowerDeliveryReq message with an invalid SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP

	connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.8.4.2; 15118-2.8.4.3.9.3; 15118-2.8.7.4.3; 15118-2.8.8.3.1; 15118-2.8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_003	

Table 273 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_010'.

Table 273 — Test case description for 'TC_SECC_AC_VTB_PowerDelivery_010'

TC Id	TC_SECC_AC_VTB_PowerDelivery_010
Test objective	Test System executes GoodCase procedure, sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', invalid charging profile (>EVSEMaximumPowerLimit), valid TupleID and all additional mandatory parameters and signals CP State C or D. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ChargingProfileInvalid' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.8.4.2; 15118-2.8.4.3.9.3; 15118-2.8.7.4.3; 15118-2.8.8.3.1; 15118-2.8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-225], [V2G2-478], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_004	

Table 274 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_011'.

Table 274 — Test case description for 'TC_SECC_AC_VTB_PowerDelivery_011'

TC Id	TC_SECC_AC_VTB_PowerDelivery_011
Test objective	Test System executes GoodCase procedure, sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', valid charging profile, invalid TupleID and all additional mandatory parameters and signals CP State C or D. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_TariffSelectionInvalid' and all additional

	mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.9.3; 15118-2:8.7.4.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-479], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_005	

Table 275 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_012'.

Table 275 — Test case description for 'TC_SECC_AC_VTB_PowerDelivery_012'

TC Id	TC_SECC_AC_VTB_PowerDelivery_012
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PowerDelivery_001	

Table 276 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_013'.

Table 276 — Test case description for 'TC_SECC_AC_VTB_PowerDelivery_013'

TC Id	TC_SECC_AC_VTB_PowerDelivery_013
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a valid ChargingStatusRes or MeteringReceipt message. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3

Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PowerDelivery_001	

Table 277 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_014'.

Table 277 — Test case description for 'TC_SECC_AC_VTB_PowerDelivery_014'

TC Id	TC_SECC_AC_VTB_PowerDelivery_014
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters but no PowerDeliveryReq with ChargeProgress 'Start' was sent before. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.9.3; 15118-2:8.7.4.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-458], [V2G2-812]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_006	

10.2.2.8.2 DC specific test cases

Table 278 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_001'.

Table 278 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_001'

TC Id	TC_SECC_DC_VTB_PowerDelivery_001
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', valid charging profile and TupleID and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1;

	15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-224], [V2G2-590], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PreCharge_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 279 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_002'.

Table 279 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_002'

TC Id	TC_SECC_DC_VTB_PowerDelivery_002
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop' and all additional mandatory parameters. Furthermore Test System signals CP State B after received valid PowerDeliveryRes message. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 280 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_003'.

Table 280 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_003'

TC Id	TC_SECC_DC_VTB_PowerDelivery_003
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a PowerDeliveryReq message with ChargeProgressType 'Start_'. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.3.3; 15118-2:8.8.3.1; 15118-2:8.8.4.2.3; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PreCharge_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_002	

Table 281 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_004'.

Table 281 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_004'

TC Id	TC_SECC_DC_VTB_PowerDelivery_004
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a PowerDeliveryReq message with ChargeProgressType 'Stop_'. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.3.3; 15118-2:8.8.3.1; 15118-2:8.8.4.2.3; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_002	

Table 282 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_005'.

Table 282 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_005'

TC Id	TC_SECC_DC_VTB_PowerDelivery_005
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with an invalid SessionID, ChargeProgress 'Start_', valid charging profile and TupleID and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.3.1; 15118-2:8.8.4.2.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PreCharge_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_003	

Table 283 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_006'.

Table 283 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_006'

TC Id	TC_SECC_DC_VTB_PowerDelivery_006
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with an invalid SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.3.1; 15118-2:8.8.4.2.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_003	

Table 284 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_007'.

Table 284 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_007'

TC Id	TC_SECC_DC_VTB_PowerDelivery_007
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', invalid charging profile (>EVSEMaximumPowerLimit), valid TupleID and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ChargingProfileInvalid' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.3.1; 15118-2:8.8.4.2.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-225], [V2G2-478], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PreCharge_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_004	

Table 285 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_008'.

Table 285 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_008'

TC Id	TC_SECC_DC_VTB_PowerDelivery_008
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', valid charging profile, invalid TupleID and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_TariffSelectionInvalid' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.3.1; 15118-2:8.8.4.2.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-479], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PreCharge_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_005	

Table 286 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_009'.

Table 286 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_009'

TC Id	TC_SECC_DC_VTB_PowerDelivery_009
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS

	Section(s): 15118-2.7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PreCharge_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PowerDelivery_001	

Table 287 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_010'.

Table 287 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_010'

TC Id	TC_SECC_DC_VTB_PowerDelivery_010
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a valid CurrentDemandRes or MeteringReceipt message. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PowerDelivery_001	

Table 288 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_011'.

Table 288 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_011'

TC Id	TC_SECC_DC_VTB_PowerDelivery_011
Test objective	Test System executes GoodCase procedure and initiates a renegotiation by sending a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-813], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665]

Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 289 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_012'.

Table 289 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_012'

TC Id	TC_SECC_DC_VTB_PowerDelivery_012
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_' and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.7.4.3; 15118-2:8.8.4.3.2; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-590], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 290 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_013'.

Table 290 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_013'

TC Id	TC_SECC_DC_VTB_PowerDelivery_013
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters but no PowerDeliveryReq with ChargeProgress 'Start' was sent before. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS

	Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.9.3; 15118-2:8.7.4.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-458], [V2G2-812]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PreCharge_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_006	

Table 291 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_014'

Table 291 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_014'

TC Id	TC_SECC_DC_VTB_PowerDelivery_014
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, EVErrorcode 'FAILED_RESSTemperatureInhibit', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorcode not influences the EVSE charging process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.3.3; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 292 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_015'.

Table 292 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_015'

TC Id	TC_SECC_DC_VTB_PowerDelivery_015
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, EVErrorcode 'FAILED_EVShiftPosition', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorcode not influences the EVSE charging process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.3.3; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 293 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_016'.

Table 293 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_016'

TC Id	TC_SECC_DC_VTB_PowerDelivery_016
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, EVErrorcode 'FAILED_ChargerConnectorLockFault', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorcode not influences the EVSE charging process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.3.3; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 294 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_017'.

Table 294 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_017'

TC Id	TC_SECC_DC_VTB_PowerDelivery_017
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, EVErrorcode 'FAILED_EVRESSMalfunction', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorcode not influences the EVSE charging

	process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.3.3; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 295 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_018'.

Table 295 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_018'

TC Id	TC_SECC_DC_VTB_PowerDelivery_018
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, EVErrorcode 'FAILED_ChargingCurrentdifferential', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorcode not influences the EVSE charging process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.3.3; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 296 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_019'.

Table 296 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_019'

TC Id	TC_SECC_DC_VTB_PowerDelivery_019
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message

	with the current SessionID, EVErrorcode 'FAILED_ChargingVoltageOutOfRange', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorcode not influences the EVSE charging process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.3.3; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 297 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_020'.

Table 297 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_020'

TC Id	TC_SECC_DC_VTB_PowerDelivery_020
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, EVErrorcode 'FAILED_ChargingSystemIncompatibility', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorcode not influences the EVSE charging process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.3.3; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 298 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_021'.

Table 298 — Test case description for 'TC_SECC_DC_VTB_PowerDelivery_021'

TC Id	TC_SECC_DC_VTB_PowerDelivery_021
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, EVErrorcode 'NoData', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorcode not influences the EVSE charging process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.3.3; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

10.2.2.9 SECC test cases for CertificateUpdate

The following subclauses include all *V2G CertificateUpdate* message test case descriptions, where the SECC is defined as SUT.

10.2.2.9.1 Common test cases

Table 299 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_001'.

Table 299 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_001'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_001
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, contract certificate chain, EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with a valid signature, current SessionID, ResponseCode 'OK', SProvisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.10.3; 15118-2:8.7.3.1; 15118-2:8.8.4.3; 15118-2:8.5.2.28; 15118-2:8.5.2.29; 15118-2:8.5.2.30; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.4; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.9.2.4.5; 15118-2:8.5.2.26; 15118-2:7.9.2.4.3; 15118-2:7.3.2; 15118-2:7.3.3; 15118-2:AnnexF; 15118-2:7.9.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-274], [V2G2-275], [V2G2-557], [V2G2-752], [V2G2-778], [V2G2-779], [V2G2-780], [V2G2-781], [V2G2-782], [V2G2-783], [V2G2-389], [V2G2-401], [V2G2-412], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-653], [V2G2-353], [V2G2-354], [V2G2-814], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-

	926], [V2G2-884], [V2G2-822], [V2G2-665], [V2G2-103], [V2G2-104]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_001	

Table 300 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_002'.

Table 300 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_002'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a CertificateUpdateReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.3.3.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-557], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_002	

Table 301 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_003'.

Table 301 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_003'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_003
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, invalid SessionID, contract certificate chain, valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.3.10.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001

PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_003	

Table 302 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_004'.

Table 302 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_004'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_004
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_004	

Table 303 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_005'.

Table 303 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_005'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_005
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with an invalid signature (wrong private key used), current SessionID, valid EMAID contract certificate chain, valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2.7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-461], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1

PreCondition
f_SECC_CMN_PR_PaymentServiceSelection_001
Expected behavior
f_SECC_CMN_TB_VTB_CertificateUpdate_005

Table 304 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_006'.

Table 304 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_006'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_006
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with an invalid signature (signed body changed after signature calculation), current SessionID, contract certificate chain, valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2.7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-461], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_006	

Table 305 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_007'.

Table 305 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_007'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_007
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, invalid contract certificate chain (Order of Sub CAs is inverted), valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED_CertChainError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2.7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-470], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC,

	PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_007	

Table 306 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_008'.

Table 306 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_008'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_008
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, invalid contract certificate chain (Leaf Certificate from wrong PKI is selected), valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED_CertChainError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-470], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_007	

Table 307 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_009'.

Table 307 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_009'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_009
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, invalid contract certificate chain (Sub2 Certificate from wrong PKI is selected), valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED_CertChainError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-470], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]

Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_007	

Table 308 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_010'.

Table 308 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_010'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_010
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, invalid contract certificate chain (Sub1 Certificate from wrong PKI is selected), valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED_CertChainError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-470], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_007	

Table 309 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_011'.

Table 309 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_011'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_011
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, invalid contract certificate chain (Certificate chain from wrong PKI is selected), valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED_CertChainError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS

	Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-470], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_007	

Table 310 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_012'.

Table 310 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_012'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_012
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, contract certificate chain, wrong EMAID (BodyType CertificateUpdateReq) and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED', 'FAILED_NoCertificateAvailable' or 'FAILED_ContractCanceled' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_008	

Table 311 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_013'.

Table 311 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_013'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_013
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, contract certificate chain, invalid EMAID (BodyType CertificateUpdateReq) and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED', 'FAILED_NoCertificateAvailable' or

	'FAILED_ContractCanceled' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2.7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_008	

Table 312 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_014'.

Table 312 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_014'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_014
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, contract certificate chain, wrong EMAID (contract certificate) and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED', 'FAILED_NoCertificateAvailable' or 'FAILED_ContractCanceled' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2.7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_009	

Table 313 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_015'.

Table 313 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_015'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_015
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, contract certificate chain, invalid EMAID

	(contract certificate) and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED', 'FAILED_NoCertificateAvailable' or 'FAILED_ContractCanceled' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2.7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_009	

Table 314 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_016'.

Table 314 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_016'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_016
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, invalid contract certificate chain (Expired), valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED_CertificateExpired' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2.7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-473], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_010	

Table 315 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_017'.

Table 315 — Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_017'

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_017
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, invalid contract certificate chain (Invalid content -> Country name was set), valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED' or 'FAILED_NoCertificateAvailable' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC PICS_CertificateHandling := certUpdate, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_010	

10.2.2.10 SECC test cases for CertificateInstallation

The following subclauses include all V2G *CertificateInstallation* message test case descriptions, where the SECC is defined as SUT.

10.2.2.10.1 Common test cases

Table 316 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_001'.

Table 316 — Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_001'

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_001
Test objective	Test System executes GoodCase procedure and sends a CertificateInstallationReq message with a valid signature, current SessionID, OEMProvisioning certificate and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateInstallationRes message with a valid signature, current SessionID, ResponseCode 'OK', SProvisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.4.3; 15118-2:8.5.2.28; 15118-2:8.5.2.29; 15118-2:8.5.2.30; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.3; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.9.2.4.5; 15118-2:8.5.2.26; 15118-2:7.9.2.4.3; 15118-2:7.3.2; 15118-2:7.3.3; 15118-2:AnnexF; 15118-2:7.9.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-237], [V2G2-238], [V2G2-274], [V2G2-275], [V2G2-554], [V2G2-752], [V2G2-778], [V2G2-779], [V2G2-780], [V2G2-781], [V2G2-782], [V2G2-783], [V2G2-387], [V2G2-399], [V2G2-410], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-653], [V2G2-353], [V2G2-354], [V2G2-814], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-

	926], [V2G2-884], [V2G2-822], [V2G2-665], [V2G2-103], [V2G2-104]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certInstall, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_001	

Table 317 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_002'.

Table 317 — Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_002'

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a CertificateInstallationReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.3.3.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certInstall, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_002	

Table 318 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_003'.

Table 318 — Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_003'

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_003
Test objective	Test System executes GoodCase procedure and sends a CertificateInstallationReq message with a valid signature, invalid SessionID, OEMProvisioning certificate and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateInstallationRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS @url ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3

Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-237], [V2G2-238], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certInstall, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_003	

Table 319 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_004'.

Table 319 — Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_004'

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_004
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.8.4.3.1; 15118-2.8.7.2.1; 15118-2.8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certInstall, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_004	

Table 320 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_005'.

Table 320 — Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_005'

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_005
Test objective	Test System executes GoodCase procedure and sends a CertificateInstallationReq message with an invalid signature (wrong private key used), current SessionID, OEMProvisioning certificate and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateInstallationRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.7.4; 15118-2.8.4.3.11.3; 15118-2.8.7.3.1; 15118-2.8.8.3.1; 15118-2.8.8.4.3
Referenced	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-237], [V2G2-238], [V2G2-

requirement(s)	752], [V2G2-461], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certInstall, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_005	

Table 321 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_006'.

Table 321 — Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_006'

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_006
Test objective	Test System executes GoodCase procedure and sends a CertificateInstallationReq message with an invalid signature (signed body changed after signature calculation), current SessionID, OEMProvisioning certificate and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateInstallationRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-237], [V2G2-238], [V2G2-752], [V2G2-461], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certInstall, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_006	

Table 322 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_007'.

Table 322 — Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_007'

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_007
Test objective	Test System executes GoodCase procedure and sends a CertificateInstallationReq message with a valid signature, current SessionID, invalid OEMProvisioning certificate (Expired) and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateInstallationRes message with the current SessionID, ResponseCode 'FAILED_CertificateExpired' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-237], [V2G2-238], [V2G2-

requirement(s)	752], [V2G2-468], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certInstall, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_007	

Table 323 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_008'.

Table 323 — Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_008'

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_008
Test objective	Test System executes GoodCase procedure and sends a CertificateInstallationReq message with a valid signature, current SessionID, invalid OEMProvisioning certificate (Invalid content -> Domain component changed) and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateInstallationRes message with the current SessionID, ResponseCode 'FAILED' or 'FAILED_NoCertificateAvailable' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.3.11.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-237], [V2G2-238], [V2G2-752], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CertificateHandling := certInstall, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_007	

10.2.2.11 SECC test cases for SessionStop

The following subclauses include all *V2G SessionStop message* test case descriptions, where the SECC is defined as SUT.

10.2.2.11.1 AC specific test cases

Table 324 lists the test case description for 'TC_SECC_AC_VTB_SessionStop_001'.

Table 324 — Test case description for 'TC_SECC_AC_VTB_SessionStop_001'

TC Id	TC_SECC_AC_VTB_SessionStop_001

Test objective	Test System executes GoodCase procedure and sends a SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Test System then checks that the SUT sends a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the data link connection within 'par_TP_match_leave'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.12.3; 15118-2:8.8.4.3.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-240], [V2G2-241], [V2G2-571], [V2G2-751], [V2G2-752], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-724], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_001	

Table 325 lists the test case description for 'TC_SECC_AC_VTB_SessionStop_002'.

Table 325 — Test case description for 'TC_SECC_AC_VTB_SessionStop_002'

TC Id	TC_SECC_AC_VTB_SessionStop_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a SessionStopReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.3.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_002	

Table 326 lists the test case description for 'TC_SECC_AC_VTB_SessionStop_003'.

Table 326 — Test case description for 'TC_SECC_AC_VTB_SessionStop_003'

TC Id	TC_SECC_AC_VTB_SessionStop_003

Test objective	Test System executes GoodCase procedure and sends a SessionStopReq message with an invalid SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Test System then checks that the SUT sends a SessionStopRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.8.4.2; 15118-2.8.4.3.12.3; 15118-2.8.8.3.1; 15118-2.8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-240], [V2G2-241], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_003	

Table 327 lists the test case description for 'TC_SECC_AC_VTB_SessionStop_004'.

Table 327 — Test case description for 'TC_SECC_AC_VTB_SessionStop_004'

TC Id	TC_SECC_AC_VTB_SessionStop_004
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.8.4.3.1; 15118-2.8.7.2.1; 15118-2.8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_004	

10.2.2.11.2 DC specific test cases

Table 328 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_001'.

Table 328 — Test case description for 'TC_SECC_DC_VTB_SessionStop_001'

TC Id	TC_SECC_DC_VTB_SessionStop_001
Test objective	Test System executes GoodCase procedure and sends a SessionStopReq message with

	the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Test System then checks that the SUT sends a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the data link connection within 'par_TP_match_leave'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.12.3; 15118-2:8.8.4.3.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-240], [V2G2-241], [V2G2-571], [V2G2-751], [V2G2-752], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-568], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-921], [V2G2-724], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := false
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_001	

Table 329 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_002'.

Table 329 — Test case description for 'TC_SECC_DC_VTB_SessionStop_002'

TC Id	TC_SECC_DC_VTB_SessionStop_002
Test objective	Test System executes GoodCase procedure and sends a SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Test System then checks that the SUT sends a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the data link connection within 'par_TP_match_leave'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.12.3; 15118-2:8.8.4.3.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-240], [V2G2-241], [V2G2-571], [V2G2-751], [V2G2-752], [V2G2-391], [V2G2-395], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-921], [V2G2-724], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_WeldingDetection_001	
Expected behavior	

f_SECC_CMN_TB_VTB_SessionStop_001

Table 330 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_003'.

Table 330 — Test case description for 'TC_SECC_DC_VTB_SessionStop_003'

TC Id	TC_SECC_DC_VTB_SessionStop_003
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a SessionStopReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.3.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := false
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_002	

Table 331 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_004'.

Table 331 — Test case description for 'TC_SECC_DC_VTB_SessionStop_004'

TC Id	TC_SECC_DC_VTB_SessionStop_004
Test objective	Test System executes GoodCase procedure and sends a SessionStopReq message with an invalid SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Test System then checks that the SUT sends a SessionStopRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.12.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-240], [V2G2-241], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := false
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	

f_SECC_CMN_TB_VTB_SessionStop_003

Table 332 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_005'.

Table 332 — Test case description for 'TC_SECC_DC_VTB_SessionStop_005'

TC Id	TC_SECC_DC_VTB_SessionStop_005
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := false
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_004	

Table 333 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_006'.

Table 333 — Test case description for 'TC_SECC_DC_VTB_SessionStop_006'

TC Id	TC_SECC_DC_VTB_SessionStop_006
Test objective	Test System executes GoodCase procedure and sends a SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters without signaling CP State B before. Test System then checks that the SUT sends a SessionStopRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.12.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-240], [V2G2-241], [V2G2-752], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-458], [V2G2-922], [V2G2-919], [V2G2-920]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := false
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_005	

Table 334 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_007'.

Table 334 — Test case description for 'TC_SECC_DC_VTB_SessionStop_007'

TC Id	TC_SECC_DC_VTB_SessionStop_007
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a SessionStopReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.8.4.2; 15118-2.8.4.3.3.3; 15118-2.8.8.3.1; 15118-2.8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_WeldingDetection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_002	

Table 335 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_008'.

Table 335 — Test case description for 'TC_SECC_DC_VTB_SessionStop_008'

TC Id	TC_SECC_DC_VTB_SessionStop_008
Test objective	Test System executes GoodCase procedure and sends a SessionStopReq message with an invalid SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Test System then checks that the SUT sends a SessionStopRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.8.4.2; 15118-2.8.4.3.12.3; 15118-2.8.8.3.1; 15118-2.8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-240], [V2G2-241], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_WeldingDetection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_003	

Table 336 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_009'.

Table 336 — Test case description for 'TC_SECC_DC_VTB_SessionStop_009'

TC Id	TC_SECC_DC_VTB_SessionStop_009
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_WeldingDetection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_004	

10.2.2.12 SECC test cases for ChargingStatus

The following subclause includes all *V2G ChargingStatus message* test case descriptions, where the SECC is defined as SUT.

10.2.2.12.1 AC specific test cases

Table 337 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_001'.

Table 337 — Test case description for 'TC_SECC_AC_VTB_ChargingStatus_001'

TC Id	TC_SECC_AC_VTB_ChargingStatus_001
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.4.2.3; 15118-2:8.6.3.2; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-575], [V2G2-278], [V2G2-279], [V2G2-752], [V2G2-577], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-407], [V2G2-691], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := false, PIXIT_SECC_CMN_MeterInfo := unknown

PreCondition
f_SECC_AC_PR_PowerDeliveryStart_001
Expected behavior
f_SECC_AC_TB_VTB_ChargingStatus_001

Table 338 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_002'.

Table 338 — Test case description for 'TC_SECC_AC_VTB_ChargingStatus_002'

TC Id	TC_SECC_AC_VTB_ChargingStatus_002
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.4.2.3; 15118-2:8.6.3.2; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-575], [V2G2-278], [V2G2-279], [V2G2-752], [V2G2-577], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-406], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_001	

Table 339 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_003'.

Table 339 — Test case description for 'TC_SECC_AC_VTB_ChargingStatus_003'

TC Id	TC_SECC_AC_VTB_ChargingStatus_003
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a ChargingStatusReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.3.3; 15118-2:8.6.3.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.2; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001

PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_002	

Table 340 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_004'.

Table 340 — Test case description for 'TC_SECC_AC_VTB_ChargingStatus_004'

TC Id	TC_SECC_AC_VTB_ChargingStatus_004
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with an invalid SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.4.2.3, 15118-2:8.6.3.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_003	

Table 341 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_005'.

Table 341 — Test case description for 'TC_SECC_AC_VTB_ChargingStatus_005'

TC Id	TC_SECC_AC_VTB_ChargingStatus_005
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	

f_SECC_AC_PR_PowerDeliveryStart_001
Expected behavior
f_SECC_AC_TB_VTB_ChargingStatus_004

Table 342 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_006'.

Table 342 — Test case description for 'TC_SECC_AC_VTB_ChargingStatus_006'

TC Id	TC_SECC_AC_VTB_ChargingStatus_006
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false', a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.4.2.3; 15118-2:8.6.3.2; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-575], [V2G2-278], [V2G2-279], [V2G2-752], [V2G2-577], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-407], [V2G2-691], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665], [V2G2-276], [V2G2-277]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := false, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_005	

Table 343 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_007'.

Table 343 — Test case description for 'TC_SECC_AC_VTB_ChargingStatus_007'

TC Id	TC_SECC_AC_VTB_ChargingStatus_007
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true', a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.4.2.3; 15118-2:8.6.3.2; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-575], [V2G2-278], [V2G2-279], [V2G2-752], [V2G2-577], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-406], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665], [V2G2-276], [V2G2-277]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_PkiName := PKI_1

PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_005	

Table 344 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_008'.

Table 344 — Test case description for 'TC_SECC_AC_VTB_ChargingStatus_008'

TC Id	TC_SECC_AC_VTB_ChargingStatus_008
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false', a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.4.2.3, 15118-2:8.6.3.2; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-575], [V2G2-278], [V2G2-279], [V2G2-752], [V2G2-577], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-407], [V2G2-691], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665], [V2G2-276], [V2G2-277], [V2G2-831]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := false, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := meterReading
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_005	

Table 345 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_009'.

Table 345 — Test case description for 'TC_SECC_AC_VTB_ChargingStatus_009'

TC Id	TC_SECC_AC_VTB_ChargingStatus_009
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true', a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.4.2.3, 15118-2:8.6.3.2; 15118-2:8.8.4.3.2; 15118-2:8.5.2.7; 15118-2:8.5.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.5.2.6
Referenced	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-575], [V2G2-278], [V2G2-279], [V2G2-752], [V2G2-577], [V2G2-358], [V2G2-359],

requirement(s)	[V2G2-381], [V2G2-406], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665], [V2G2-276], [V2G2-277], [V2G2-831]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := meterReading
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_005	

10.2.2.13 SECC test cases for MeteringReceipt

The following subclauses include all *V2G MeteringReceipt message* test case descriptions, where the SECC is defined as SUT.

10.2.2.13.1 AC specific test cases

Table 346 lists the test case description for 'TC_SECC_AC_VTB_MeteringReceipt_001'.

Table 346 — Test case description for 'TC_SECC_AC_VTB_MeteringReceipt_001'

TC Id	TC_SECC_AC_VTB_MeteringReceipt_001
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.3; 15118-2:8.4.3.2; 15118-2:8.4.2; 15118-2:8.5.3.1; 15118-2:8.6.2.3.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.13.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-580], [V2G2-752], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-005], [V2G2-007], [V2G2-653], [V2G2-665], [V2G2-904]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_AC_PR_ChargingStatus_001	
Expected behavior	
f_SECC_AC_TB_VTB_MeteringReceipt_001	

Table 347 lists the test case description for 'TC_SECC_AC_VTB_MeteringReceipt_002'.

Table 347 — Test case description for 'TC_SECC_AC_VTB_MeteringReceipt_002'

TC Id	TC_SECC_AC_VTB_MeteringReceipt_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a MeteringReceiptReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.3; 15118-2:8.4.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_AC_PR_ChargingStatus_001	
Expected behavior	
f_SECC_CMN_TB_VTB_MeteringReceipt_001	

Table 348 lists the test case description for "TC_SECC_AC_VTB_MeteringReceipt_003".

Table 348 — Test case description for "TC_SECC_AC_VTB_MeteringReceipt_003"

TC Id	TC_SECC_AC_VTB_MeteringReceipt_003
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with a valid signature, invalid sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.3; 15118-2:8.4.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_AC_PR_ChargingStatus_001	
Expected behavior	
f_SECC_AC_TB_VTB_MeteringReceipt_002	

Table 349 lists the test case description for 'TC_SECC_AC_VTB_MeteringReceipt_004'.

Table 349 — Test case description for 'TC_SECC_AC_VTB_MeteringReceipt_004'

TC Id	TC_SECC_AC_VTB_MeteringReceipt_004
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, invalid MeterReading and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.8.4.3.13.3; 15118-2.8.4.2; 15118-2.8.8.3.1; 15118-2.8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-481], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_AC_PR_ChargingStatus_001	
Expected behavior	
f_SECC_AC_TB_VTB_MeteringReceipt_003	

Table 350 lists the test case description for 'TC_SECC_AC_VTB_MeteringReceipt_005'.

Table 350 — Test case description for 'TC_SECC_AC_VTB_MeteringReceipt_005'

TC Id	TC_SECC_AC_VTB_MeteringReceipt_005
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.8.4.3.1; 15118-2.8.7.2.1; 15118-2.8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_AC_PR_ChargingStatus_001	

Expected behavior
f_SECC_CMN_TB_VTB_MeteringReceipt_002

Table 351 lists the test case description for 'TC_SECC_AC_VTB_MeteringReceipt_006'.

Table 351 — Test case description for 'TC_SECC_AC_VTB_MeteringReceipt_006'

TC Id	TC_SECC_AC_VTB_MeteringReceipt_006
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with an invalid signature (wrong private key used), current sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.3; 15118-2:8.4.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-481], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := ac, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_AC_PR_ChargingStatus_001	
Expected behavior	
f_SECC_AC_TB_VTB_MeteringReceipt_004	

Table 352 lists the test case description for 'TC_SECC_AC_VTB_MeteringReceipt_007'.

Table 352 — Test case description for 'TC_SECC_AC_VTB_MeteringReceipt_007'

TC Id	TC_SECC_AC_VTB_MeteringReceipt_007
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with an invalid signature (signed body changed after signature calculation), current sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.3; 15118-2:8.4.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-481], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := ac, PICS_CMN_CMN_PkiName := PKI_1

PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_AC_PR_ChargingStatus_001	
Expected behavior	
f_SECC_AC_TB_VTB_MeteringReceipt_005	

10.2.2.13.2 DC specific test cases

Table 353 lists the test case description for 'TC_SECC_DC_VTB_MeteringReceipt_001'.

Table 353 — Test case description for 'TC_SECC_DC_VTB_MeteringReceipt_001'

TC Id	TC_SECC_DC_VTB_MeteringReceipt_001
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.3; 15118-2:8.4.3.3; 15118-2:8.4.2; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.3.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.13.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-797], [V2G2-752], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-005], [V2G2-007], [V2G2-653], [V2G2-665], [V2G2-904]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_DC_PR_CurrentDemand_001	
Expected behavior	
f_SECC_DC_TB_VTB_MeteringReceipt_001	

Table 354 lists the test case description for 'TC_SECC_DC_VTB_MeteringReceipt_002'.

Table 354 — Test case description for 'TC_SECC_DC_VTB_MeteringReceipt_002'

TC Id	TC_SECC_DC_VTB_MeteringReceipt_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a MeteringReceiptReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.3.3; 15118-2:8.4.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-

requirement(s)	752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_DC_PR_CurrentDemand_001	
Expected behavior	
f_SECC_CMN_TB_VTB_MeteringReceipt_001	

Table 355 lists the test case description for 'TC_SECC_DC_VTB_MeteringReceipt_003'.

Table 355 — Test case description for 'TC_SECC_DC_VTB_MeteringReceipt_003'

TC Id	TC_SECC_DC_VTB_MeteringReceipt_003
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with a valid signature, invalid sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.3; 15118-2:8.4.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_DC_PR_CurrentDemand_001	
Expected behavior	
f_SECC_DC_TB_VTB_MeteringReceipt_002	

Table 356 lists the test case description for 'TC_SECC_DC_VTB_MeteringReceipt_004'.

Table 356 — Test case description for 'TC_SECC_DC_VTB_MeteringReceipt_004'

TC Id	TC_SECC_DC_VTB_MeteringReceipt_004
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, invalid MeterReading and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.8.4.3.13.3; 15118-2.8.4.2; 15118-2.8.8.3.1; 15118-2.8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-481], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_DC_PR_CurrentDemand_001	
Expected behavior	
f_SECC_DC_TB_VTB_MeteringReceipt_003	

Table 357 lists the test case description for 'TC_SECC_DC_VTB_MeteringReceipt_005'.

Table 357 — Test case description for 'TC_SECC_DC_VTB_MeteringReceipt_005'

TC Id	TC_SECC_DC_VTB_MeteringReceipt_005
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false' and current and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.8.4.3.1; 15118-2.8.7.2.1; 15118-2.8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_DC_PR_CurrentDemand_001	
Expected behavior	
f_SECC_CMN_TB_VTB_MeteringReceipt_002	

Table 358 lists the test case description for 'TC_SECC_DC_VTB_MeteringReceipt_006'.

Table 358 — Test case description for 'TC_SECC_DC_VTB_MeteringReceipt_006'

TC Id	TC_SECC_DC_VTB_MeteringReceipt_006
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with an invalid signature (wrong private key used), current sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and all

	additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.3; 15118-2:8.4.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-481], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_DC_PR_CurrentDemand_001	
Expected behavior	
f_SECC_DC_TB_VTB_MeteringReceipt_004	

Table 359 lists the test case description for 'TC_SECC_DC_VTB_MeteringReceipt_007'.

Table 359 — Test case description for 'TC_SECC_DC_VTB_MeteringReceipt_007'

TC Id	TC_SECC_DC_VTB_MeteringReceipt_007
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with an invalid signature (signed body changed after signature calculation), current sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.3; 15118-2:8.4.2; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-481], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_DC_PR_CurrentDemand_001	
Expected behavior	
f_SECC_DC_TB_VTB_MeteringReceipt_005	

10.2.2.14 SECC test cases for CableCheck

The following subclause includes all *V2G CableCheck message* test case descriptions, where the SECC is defined as SUT.

10.2.2.14.1 DC specific test cases

Table 360 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_001'.

Table 360 — Test case description for 'TC_SECC_DC_VTB_CableCheck_001'

TC Id	TC_SECC_DC_VTB_CableCheck_001
Test objective	Test System executes GoodCase procedure, signals CP State C and sends a CableCheckReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. If a CableCheckRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another CableCheckReq message. This message sequence will be repeated until the cable check of the SUT has finished or 'V2G_EVCC_CableCheck_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.5.2.3; 15118-2:8.5.4.1; 15118-2:8.7.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-251], [V2G2-252], [V2G2-584], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-917], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_EvStateInChargeProgress := C
PIXIT selection	PIXIT_SECC_DC_CableCheckOngoing == unknown
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_001	

Table 361 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_002'.

Table 361 — Test case description for 'TC_SECC_DC_VTB_CableCheck_002'

TC Id	TC_SECC_DC_VTB_CableCheck_002
Test objective	Test System executes GoodCase procedure, signals CP State D and sends a CableCheckReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. If a CableCheckRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another CableCheckReq message. This message sequence will be repeated until the cable check of the SUT has finished or 'V2G_EVCC_CableCheck_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.5.2.3; 15118-2:8.5.4.1; 15118-2:8.7.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-251], [V2G2-252], [V2G2-364], [V2G2-365], [V2G2-584], [V2G2-752], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-

	677], [V2G2-917], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_EvStateInChargeProgress := D
PIXIT selection	PIXIT_SECC_DC_CableCheckOngoing == unknown
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_001	

Table 362 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_003'.

Table 362 — Test case description for 'TC_SECC_DC_VTB_CableCheck_003'

TC Id	TC_SECC_DC_VTB_CableCheck_003
Test objective	Test System executes GoodCase procedure, signals CP State C or D and sends a valid ServiceDiscoveryReq message instead of a CableCheckReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.4.3.3.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_002	

Table 363 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_004'.

Table 363 — Test case description for 'TC_SECC_DC_VTB_CableCheck_004'

TC Id	TC_SECC_DC_VTB_CableCheck_004
Test objective	Test System executes GoodCase procedure, signals CP State C or D and sends a CableCheckReq message with an invalid SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a CableCheckRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.4.5.2.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-251], [V2G2-252], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]

Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_003	

Table 364 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_005'.

Table 364 — Test case description for 'TC_SECC_DC_VTB_CableCheck_005'

TC Id	TC_SECC_DC_VTB_CableCheck_005
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.8.4.3.1; 15118-2.8.7.2.1; 15118-2.8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_004	

Table 365 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_006'.

Table 365 — Test case description for 'TC_SECC_DC_VTB_CableCheck_006'

TC Id	TC_SECC_DC_VTB_CableCheck_006
Test objective	Test System executes GoodCase procedure, signals CP State D and sends a CableCheckReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Furthermore the repetition of the CableCheck message sequence will be checked until the cable check of the SUT has finished or 'V2G_EVCC_CableCheck_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.8.4.5.2.3; 15118-2.8.5.4.1; 15118-2.8.7.3.1; 15118-2.8.8.4.3; 15118-2.8.6.2.4.1; 15118-2.8.6.2.4.2; 15118-2.8.8.3.1; 15118-2.7.3.4; 15118-2.8.6.2.1; 15118-2.8.8.4.1; 15118-2.8.7.4.4; 15118-2.8.8.4.3.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-251], [V2G2-252], [V2G2-364], [V2G2-365], [V2G2-584], [V2G2-752], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-

	677], [V2G2-917], [V2G2-915], [V2G2-665], [V2G2-621]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_EvStateInChargeProgress := C
PIXIT selection	PIXIT_SECC_DC_CableCheckOngoing == ongoing
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_001	

Table 366 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_007'.

Table 366 — Test case description for 'TC_SECC_DC_VTB_CableCheck_007'

TC Id	TC_SECC_DC_VTB_CableCheck_007
Test objective	Test System executes GoodCase procedure, signals CP State D and sends a CableCheckReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Furthermore the repetition of the CableCheck message sequence will be checked until the cable check of the SUT has finished or 'V2G_EVCC_CableCheck_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.5.2.3; 15118-2:8.5.4.1; 15118-2:8.7.3.1; 15118-2:8.8.4.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.4; 15118-2:8.8.4.3.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-251], [V2G2-252], [V2G2-364], [V2G2-365], [V2G2-584], [V2G2-752], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-917], [V2G2-915], [V2G2-665], [V2G2-621]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_EvStateInChargeProgress := D
PIXIT selection	PIXIT_SECC_DC_CableCheckOngoing == ongoing
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_001	

Table 367 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_008'.

Table 367 — Test case description for 'TC_SECC_DC_VTB_CableCheck_008'

TC Id	TC_SECC_DC_VTB_CableCheck_008
Test objective	Test System executes GoodCase procedure, signals no CP State C or D and sends a CableCheckReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a CableCheckRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.4.5.2.3; 15118-2:8.7.3.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-251], [V2G2-252], [V2G2-752], [V2G2-918], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_005	

10.2.2.15 SECC test cases for PreCharge

The following subclause includes all *V2G PreCharge message* test case descriptions, where the SECC is defined as SUT.

10.2.2.15.1 DC specific test cases

Table 368 lists the test case description for 'TC_SECC_DC_VTB_PreCharge_001'.

Table 368 — Test case description for 'TC_SECC_DC_VTB_PreCharge_001'

TC Id	TC_SECC_DC_VTB_PreCharge_001
Test objective	Test System executes GoodCase procedure and sends a PreChargeReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a PreChargeRes message with the current SessionID, ResponseCode 'OK', EVSEStatusCode 'EVSE_Ready' and all additional mandatory parameters. If the Test System sends another valid PreChargeReq message, the SUT shall respond to this request message again. This message sequence will be repeated until the required voltage has been reached or 'V2G_EVCC_PreCharge_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.5.3; 15118-2:8.7.3.7; 15118-2:8.5.2.7; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-255], [V2G2-256], [V2G2-278], [V2G2-279], [V2G2-587], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CableCheck_001	
Expected behavior	
f_SECC_DC_TB_VTB_PreCharge_001	

Table 369 lists the test case description for 'TC_SECC_DC_VTB_PreCharge_002'.

Table 369 — Test case description for 'TC_SECC_DC_VTB_PreCharge_002'

TC Id	TC_SECC_DC_VTB_PreCharge_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a PreChargeReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.3.3; 15118-2:8.7.3.7; 15118-2:8.5.4.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CableCheck_001	
Expected behavior	
f_SECC_DC_TB_VTB_PreCharge_002	

Table 370 lists the test case description for 'TC_SECC_DC_VTB_PreCharge_003'.

Table 370 — Test case description for 'TC_SECC_DC_VTB_PreCharge_003'

TC Id	TC_SECC_DC_VTB_PreCharge_003
Test objective	Test System executes GoodCase procedure and sends a PreChargeReq message with an invalid SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a PreChargeRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.5.3; 15118-2:8.7.3.7; 15118-2:8.5.4.1; 15118-2:8.8.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-255], [V2G2-256], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CableCheck_001	
Expected behavior	
f_SECC_DC_TB_VTB_PreCharge_003	

Table 371 lists the test case description for 'TC_SECC_DC_VTB_PreCharge_004'.

Table 371 — Test case description for 'TC_SECC_DC_VTB_PreCharge_004'

TC Id	TC_SECC_DC_VTB_PreCharge_004

Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CableCheck_001	
Expected behavior	
f_SECC_DC_TB_VTB_PreCharge_004	

10.2.2.16 SECC test cases for CurrentDemand

The following subclause includes all *V2G CurrentDemand* message test case descriptions, where the SECC is defined as SUT.

10.2.2.16.1 DC specific test cases

Table 372 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_001'.

Table 372 — Test case description for 'TC_SECC_DC_VTB_CurrentDemand_001'

TC Id	TC_SECC_DC_VTB_CurrentDemand_001
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.5.4.3; 15118-2:8.5.4.1; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.2; 15118-2:8.6.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-278], [V2G2-279], [V2G2-259], [V2G2-260], [V2G2-593], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-788], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := false, PIXIT_SECC_CMN_MeterInfo := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	

Expected behavior
f_SECC_DC_TB_VTB_CurrentDemand_001

Table 373 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_002'.

Table 373 — Test case description for 'TC_SECC_DC_VTB_CurrentDemand_002'

TC Id	TC_SECC_DC_VTB_CurrentDemand_002
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.5.4.3; 15118-2:8.5.4.1; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-278], [V2G2-279], [V2G2-259], [V2G2-260], [V2G2-593], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-395], [V2G2-787], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-795], [V2G2-915], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dc, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_001	

Table 374 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_003'.

Table 374 — Test case description for 'TC_SECC_DC_VTB_CurrentDemand_003'

TC Id	TC_SECC_DC_VTB_CurrentDemand_003
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a CurrentDemandReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.3.3; 15118-2:8.5.4.1; 15118-2:8.8.3.1; 15118-2:8.8.4.3.3; 15118-2:8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dc

PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_002	

Table 375 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_004'.

Table 375 — Test case description for 'TC_SECC_DC_VTB_CurrentDemand_004'

TC Id	TC_SECC_DC_VTB_CurrentDemand_004
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.5.4.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-259], [V2G2-260], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_003	

Table 376 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_005'.

Table 376 — Test case description for 'TC_SECC_DC_VTB_CurrentDemand_005'

TC Id	TC_SECC_DC_VTB_CurrentDemand_005
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	

f_SECC_DC_PR_PowerDeliveryStart_001
Expected behavior
f_SECC_DC_TB_VTB_CurrentDemand_004

Table 377 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_006'.

Table 377 — Test case description for 'TC_SECC_DC_VTB_CurrentDemand_006'

TC Id	TC_SECC_DC_VTB_CurrentDemand_006
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false', a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.5.4.3; 15118-2:8.5.4.1; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.2; 15118-2:8.6.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-278], [V2G2-279], [V2G2-259], [V2G2-260], [V2G2-593], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-788], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-276], [V2G2-277]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dc
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := false, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_005	

Table 378 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_007'.

Table 378 — Test case description for 'TC_SECC_DC_VTB_CurrentDemand_007'

TC Id	TC_SECC_DC_VTB_CurrentDemand_007
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true', a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.5.4.3; 15118-2:8.5.4.1; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-278], [V2G2-279], [V2G2-259], [V2G2-260], [V2G2-593], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-395], [V2G2-787], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-

	667], [V2G2-795], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-276], [V2G2-277]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_005	

Table 379 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_008'.

Table 379 — Test case description for 'TC_SECC_DC_VTB_CurrentDemand_008'

TC Id	TC_SECC_DC_VTB_CurrentDemand_008
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false', a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.5.4.3; 15118-2:8.5.4.1; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.2; 15118-2:8.6.3.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-278], [V2G2-279], [V2G2-259], [V2G2-260], [V2G2-593], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-788], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-276], [V2G2-277], [V2G2-831]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := false, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := meterReading
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_005	

Table 380 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_009'.

Table 380 — Test case description for 'TC_SECC_DC_VTB_CurrentDemand_009'

TC Id	TC_SECC_DC_VTB_CurrentDemand_009
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with the

	current SessionID, ResponseCode 'OK', ReceiptRequired 'true', a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.5.4.3; 15118-2:8.5.4.1; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.2; 15118-2:8.8.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-278], [V2G2-279], [V2G2-259], [V2G2-260], [V2G2-593], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-395], [V2G2-787], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-795], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-276], [V2G2-277], [V2G2-831]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := true, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := meterReading
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_005	

10.2.2.17 SECC test cases for WeldingDetection

The following subclause includes all *V2G WeldingDetection* message test case descriptions, where the SECC is defined as SUT.

10.2.2.17.1 DC specific test cases

Table 381 lists the test case description for 'TC_SECC_DC_VTB_WeldingDetection_001'.

Table 381 — Test case description for 'TC_SECC_DC_VTB_WeldingDetection_001'

TC Id	TC_SECC_DC_VTB_WeldingDetection_001
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.5.5.3; 15118-2:8.8.4.3.3; 15118-2:8.5.2.7; 15118-2:8.5.4.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.8.3.1; 15118-2:7.3.4; 15118-2:8.6.2.1; 15118-2:8.8.4.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-263], [V2G2-264], [V2G2-278], [V2G2-279], [V2G2-597], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-921], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	

PreCondition
f_SECC_DC_PR_PowerDeliveryStop_001
Expected behavior
f_SECC_DC_TB_VTB_WeldingDetection_001

Table 382 lists the test case description for 'TC_SECC_DC_VTB_WeldingDetection_002'.

Table 382 — Test case description for 'TC_SECC_DC_VTB_WeldingDetection_002'

TC Id	TC_SECC_DC_VTB_WeldingDetection_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a WeldingDetectionReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.8.4.2; 15118-2.8.4.3.3.3; 15118-2.8.8.3.1; 15118-2.8.8.4.3.3; 15118-2.8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
	f_SECC_DC_PR_PowerDeliveryStop_001
Expected behavior	
	f_SECC_DC_TB_VTB_WeldingDetection_002

Table 383 lists the test case description for 'TC_SECC_DC_VTB_WeldingDetection_003'.

Table 383 — Test case description for 'TC_SECC_DC_VTB_WeldingDetection_003'

TC Id	TC_SECC_DC_VTB_WeldingDetection_003
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionReq message with an invalid SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a WeldingDetectionRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2.8.3.2; 15118-2.8.3.3; 15118-2.8.3.4; 15118-2.8.4.2; 15118-2.8.4.5.5.3; 15118-2.8.8.3.1; 15118-2.8.8.4.3.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-263], [V2G2-264], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	

f_SECC_DC_PR_PowerDeliveryStop_001
Expected behavior
f_SECC_DC_TB_VTB_WeldingDetection_003

Table 384 lists the test case description for 'TC_SECC_DC_VTB_WeldingDetection_004'.

Table 384 — Test case description for 'TC_SECC_DC_VTB_WeldingDetection_004'

TC Id	TC_SECC_DC_VTB_WeldingDetection_004
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.8.4.3.1; 15118-2:8.7.2.1; 15118-2:8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_WeldingDetection_004	

Table 385 lists the test case description for 'TC_SECC_DC_VTB_WeldingDetection_005'.

Table 385 — Test case description for 'TC_SECC_DC_VTB_WeldingDetection_005'

TC Id	TC_SECC_DC_VTB_WeldingDetection_005
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionReq message with the current SessionID and all additional mandatory parameters without signaling CP State B before. Test System then checks that the SUT sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.5.5.3; 15118-2:8.8.3.1; 15118-2:8.8.4.3.3; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-263], [V2G2-264], [V2G2-752], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-458], [V2G2-922], [V2G2-919], [V2G2-920]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	

Expected behavior
f_SECC_DC_TB_VTB_WeldingDetection_005

10.3 EVCC test cases

This subclause covers all V2G Application layer test case descriptions, where the *EVCC* is defined as SUT.

10.3.1 V2G protocol handshake

This subclause covers the *V2G protocol handshake* test case descriptions, where the EVCC is defined as SUT.

10.3.1.1 EVCC test cases for SupportedAppProtocol

Table 386 lists the test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_001'.

Table 386 — Test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_001'

TC Id	TC_EVCC_CMN_VTB_SupportedAppProtocol_001
Test objective	Test System executes GoodCase procedure and sends an SDP response message with 'transportProtocol' equals to '0x00'H and a valid security option, port and IP address. Test System then checks that the SUT sends a correct SupportedAppProtocolReq message with a list of valid AppProtocols including ISO namespace and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.2.1; 15118-2:8.2.2; 15118-2:8.2.3; 15118-2:8.8.4.2.1; 15118-2:7.4; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:7.10.1.7; 15118-2:8.6.2.1; 15118-2:7.7.3.4; 15118-2:7.3.3; 15118-2:7.7.3.3
Referenced requirement(s)	[V2G2-165], [V2G2-166], [V2G2-167], [V2G2-175], [V2G2-178], [V2G2-483], [V2G2-022], [V2G2-024], [V2G2-376], [V2G2-380], [V2G2-390], [V2G2-394], [V2G2-628], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-021], [V2G2-603], [V2G2-645], [V2G2-008], [V2G2-875]
Config Id	CF_04_002
PICS selection	
PIXIT selection	PIXIT_EVCC_CMN_TLS := true
PreCondition	
f_EVCC_CMN_PR_SDP_001	
Expected behavior	
f_EVCC_CMN_PR_InitConfiguration_002, f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001	

Table 387 lists the test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_002'.

Table 387 — Test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_002'

TC Id	TC_EVCC_CMN_VTB_SupportedAppProtocol_002
Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'FAILED_NoNegotiation' and all additional mandatory parameters. Test System then checks that the SUT closes the TCP connection.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.2.1
Referenced requirement(s)	[V2G2-173], [V2G2-728], [V2G2-025]

Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SupportedAppProtocol_002	

Table 388 lists the test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_003'.

Table 388 — Test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_003'

TC Id	TC_EVCC_CMN_VTB_SupportedAppProtocol_003
Test objective	Test System executes GoodCase procedure, sends a valid SDP response message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a SupportedAppProtocolReq message with a list of valid AppProtocols including ISO namespace and all additional mandatory parameters. Test System then checks that the SUT closes the TCP connection.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-484], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SupportedAppProtocol_003	

Table 389 lists the test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_004'.

Table 389 — Test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_004'

TC Id	TC_EVCC_CMN_VTB_SupportedAppProtocol_004
Test objective	Test System executes GoodCase procedure and sends an SDP response message with 'transportProtocol' equals to '0x00'H and a valid security option, port and IP address. Test System then checks that the SUT sends a correct SupportedAppProtocolReq message with a list of valid AppProtocols including ISO namespace and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.2.1; 15118-2:8.2.2; 15118-2:8.2.3; 15118-2:8.8.4.2.1; 15118-2:7.4; 15118-2:8.6.2.3.1; 15118-2:8.6.2.4.1; 15118-2:7.10.1.7; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-165], [V2G2-166], [V2G2-167], [V2G2-175], [V2G2-178], [V2G2-483], [V2G2-022], [V2G2-024], [V2G2-376], [V2G2-390], [V2G2-628], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-645], [V2G2-646]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_

PreCondition
f_EVCC_CMN_PR_SDP_001
Expected behavior
f_EVCC_CMN_PR_InitConfiguration_002, f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001

Table 390 lists the test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_005'.

Table 390 — Test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_005'

TC Id	TC_EVCC_CMN_VTB_SupportedAppProtocol_005
Test objective	Test System executes GoodCase procedure and sends an SDP response message with 'transportProtocol' equals to '0x00'H and a valid security option, port and IP address. Test System then checks that the SUT sends a correct SupportedAppProtocolReq message with a list of valid AppProtocols including ISO namespace and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.2.1; 15118-2:8.2.2; 15118-2:8.2.3; 15118-2:8.8.4.2.1; 15118-2:7.4; 15118-2:8.6.2.3.1; 15118-2:8.6.2.4.1; 15118-2:7.10.1.7; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-165], [V2G2-166], [V2G2-167], [V2G2-175], [V2G2-178], [V2G2-483], [V2G2-022], [V2G2-024], [V2G2-376], [V2G2-390], [V2G2-628], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-645]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_CMN_PR_SDP_001	
Expected behavior	
f_EVCC_CMN_PR_InitConfiguration_002, f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001	

10.3.2 V2G messages

This subclause includes all *V2G message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.1 EVCC test cases for SessionSetup

The following subclauses include all *V2G SessionSetup message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.1.1 Common test cases

Table 391 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_001'.

Table 391 — Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_001'

TC Id	TC_EVCC_CMN_VTB_SessionSetup_001
Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionSetupReq message with SessionID '0', EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.4.3.2.2; 15118-2:8.8.4.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1

Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-746], [V2G2-485], [V2G2-376], [V2G2-380], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_CombinedTesting := false
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_001	

Table 392 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_002'.

Table 392 — Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_002'

TC Id	TC_EVCC_CMN_VTB_SessionSetup_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionSetupRes message with a SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-486], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_002	

Table 393 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_003'.

Table 393 — Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_003'

TC Id	TC_EVCC_CMN_VTB_SessionSetup_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionSetupRes message with a SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-486], [V2G2-728], [V2G2-025]
Config Id	CF_04_002

PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_002	

Table 394 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_004'.

Table 394 — Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_004'

TC Id	TC_EVCC_CMN_VTB_SessionSetup_004
Test objective	Test System executes GoodCase procedure, sends a valid SupportedAppProtocolRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a SessionSetupReq message with SessionID '0', valid EvccID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-486], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_003	

Table 395 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_005'.

Table 395 — Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_005'

TC Id	TC_EVCC_CMN_VTB_SessionSetup_005
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of SessionSetupRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	

f_EVCC_CMN_PR_SessionSetup_001
Expected behavior
f_EVCC_CMN_TB_VTB_SessionSetup_004

Table 396 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_006'.

Table 396 — Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_006'

TC Id	TC_EVCC_CMN_VTB_SessionSetup_006
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a SessionSetupRes with a new SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_005	

Table 397 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_007'.

Table 397 — Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_007'

TC Id	TC_EVCC_CMN_VTB_SessionSetup_007
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a SessionSetupRes with a new SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_005	

Table 398 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_008'.

Table 398 — Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_008'

TC Id	TC_EVCC_CMN_VTB_SessionSetup_008
Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionSetupReq message with SessionID '0', EvccID equals to the MAC adress of the SUT and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.4.3.2.2; 15118-2:8.8.4.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-746], [V2G2-485], [V2G2-376], [V2G2-380], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-879]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_001	

10.3.2.1.2 AC specific test cases

Table 399 lists the test case description for 'TC_EVCC_AC_VTB_SessionSetup_001'.

Table 399 — Test case description for 'TC_EVCC_AC_VTB_SessionSetup_001'

TC Id	TC_EVCC_AC_VTB_SessionSetup_001
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.4.3.2.2; 15118-2:8.8.4.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC,

	PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_006	

Table 400 lists the test case description for 'TC_EVCC_AC_VTB_SessionSetup_002'.

Table 400 — Test case description for 'TC_EVCC_AC_VTB_SessionSetup_002'

TC Id	TC_EVCC_AC_VTB_SessionSetup_002
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.4.3.2.2; 15118-2:8.8.4.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_006	

Table 401 lists the test case description for 'TC_EVCC_AC_VTB_SessionSetup_003'.

Table 401 — Test case description for 'TC_EVCC_AC_VTB_SessionSetup_003'

TC Id	TC_EVCC_AC_VTB_SessionSetup_003
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid

	SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.4.3.2.2; 15118-2:8.8.4.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_Wakeup, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_006	

Table 402 lists the test case description for 'TC_EVCC_AC_VTB_SessionSetup_004'.

Table 402 — Test case description for 'TC_EVCC_AC_VTB_SessionSetup_004'

TC Id	TC_EVCC_AC_VTB_SessionSetup_004
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.4.3.2.2; 15118-2:8.8.4.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]
Config Id	CF_04_002

PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_PmaxOW < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_006	

10.3.2.1.3 DC specific test cases

Table 403 lists the test case description for 'TC_EVCC_DC_VTB_SessionSetup_001'.

Table 403 — Test case description for 'TC_EVCC_DC_VTB_SessionSetup_001'

TC Id	TC_EVCC_DC_VTB_SessionSetup_001
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5 % duty cycle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.4.3.2.2; 15118-2:8.8.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_006	

Table 404 lists the test case description for 'TC_EVCC_DC_VTB_SessionSetup_002'.

Table 404 — Test case description for 'TC_EVCC_DC_VTB_SessionSetup_002'

TC Id	TC_EVCC_DC_VTB_SessionSetup_002
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication

	session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.4.3.2.2; 15118-2:8.8.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_006	

Table 405 lists the test case description for 'TC_EVCC_DC_VTB_SessionSetup_003'.

Table 405 — Test case description for 'TC_EVCC_DC_VTB_SessionSetup_003'

TC Id	TC_EVCC_DC_VTB_SessionSetup_003
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.4.3.2.2; 15118-2:8.8.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]

Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_006	

Table 406 lists the test case description for 'TC_EVCC_DC_VTB_SessionSetup_004'.

Table 406 — Test case description for 'TC_EVCC_DC_VTB_SessionSetup_004'

TC Id	TC_EVCC_DC_VTB_SessionSetup_004
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.2; 15118-2:8.4.3.2.1; 15118-2:8.4.3.2.2; 15118-2:8.8.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_006	

10.3.2.2 EVCC test cases for ServiceDiscovery

The following subclauses include all *V2G ServiceDiscovery message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.2.1 Common test cases

Table 407 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_001'.

Table 407 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_001'

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a SessionSetupRes message with a new SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters. Test System then checks that the SUT sends a correct ServiceDiscoveryReq message with the current SessionID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.3.2; 15118-2:8.4.3.3.3; 15118-2:8.4.3.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-193], [V2G2-194], [V2G2-487], [V2G2-747], [V2G2-751], [V2G2-376], [V2G2-380], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_001	

Table 408 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_002'.

Table 408 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_002'

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-488], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_002	

Table 409 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_003'.

Table 409 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_003'

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-488], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_002	

Table 410 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_004'.

Table 410 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_004'

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-488], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_002	

Table 411 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_005'.

Table 411 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_005'

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_005
Test objective	Test System executes GoodCase procedure, sends a valid SessionSetupRes message and

	waits until the 'V2G_Msg_Timeout' timer has expired after receiving a ServiceDiscoveryReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-488], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_003	

Table 412 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_006'.

Table 412 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_006'

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_006
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', a valid PaymentOption and SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_004	

Table 413 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_007'.

Table 413 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_007'

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_007
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', a valid PaymentOption and SupportedEnergyTransferMode and all additional mandatory parameters.

	Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_004	

Table 414 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_008'.

Table 414 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_008'

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_008
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of ServiceDiscoveryRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_005	

10.3.2.3 EVCC test cases for ServiceDetail

The following subclauses include all *V2G ServiceDetail message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.3.1 Common test cases

Table 415 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_001'.

Table 415 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_001'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_001
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid

	SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.8.4.2.1; 15118-2:8.4.3.4.1; 15118-2:8.4.2; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.5; 15118-2:8.6.3.6; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-422], [V2G2-489], [V2G2-197], [V2G2-198], [V2G2-747], [V2G2-384], [V2G2-396], [V2G2-415], [V2G2-417], [V2G2-424], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_001	

Table 416 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_002'.

Table 416 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_002'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_002
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.8.4.2.1; 15118-2:8.4.3.4.1; 15118-2:8.4.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.4.1; 15118-2:8.6.3.5; 15118-2:8.6.3.6; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-422], [V2G2-489], [V2G2-197], [V2G2-198], [V2G2-747], [V2G2-378], [V2G2-392], [V2G2-415], [V2G2-417], [V2G2-424], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_002	

Table 417 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_003'.

Table 417 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_003'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 418 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_004'.

Table 418 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_004'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 419 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_005'.

Table 419 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_005'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a

	ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 420 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_006'.

Table 420 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_006'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 421 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_007'.

Table 421 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_007'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_007
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and

	terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 422 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_008'.

Table 422 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_008'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_008
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 423 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_009'.

Table 423 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_009'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_009
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_ServiceIDInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 424 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_010'.

Table 424 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_010'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_010
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_ServiceIDInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 425 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_011'.

Table 425 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_011'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_011
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2

Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_004	

Table 426 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_012'.

Table 426 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_012'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_012
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_004	

Table 427 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_013'.

Table 427 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_013'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_013
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of ServiceDetailRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]

Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_005	

Table 428 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_014'.

Table 428 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_014'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_014
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of ServiceDetailRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_005	

Table 429 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_015'.

Table 429 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_015'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_015
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM

PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_006	

Table 430 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_016'.

Table 430 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_016'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_016
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_006	

Table 431 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_017'.

Table 431 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_017'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_017
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	

f_EVCC_CMN_PR_ServiceDetail_001
Expected behavior
f_EVCC_CMN_TB_VTB_ServiceDetail_006

Table 432 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_018'.

Table 432 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_018'

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_018
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_006	

10.3.2.4 EVCC test cases for ServiceDetailAndPaymentSelection

The following subclauses include all V2G *ServiceDetailAndPaymentSelection* message test case descriptions, where the EVCC is defined as SUT.

10.3.2.4.1 Common test cases

Table 433 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001'.

Table 433 — Test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001'

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'Contract', valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24;

	15118-2:8.5.2.25; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_001	

Table 434 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002'.

**Table 434 — Test case description for
'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'ExternalPayment', valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_002	

Table 435 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_003'.

**Table 435 — Test case description for
'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_003'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_003

Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-492], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005	

Table 436 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_004'.

**Table 436 — Test case description for
'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_004'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-492], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005	

Table 437 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_005'.

**Table 437 — Test case description for
'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_005'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_005

Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-492], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005	

Table 438 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_006'.

**Table 438 — Test case description for
'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_006'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_PaymentSelectionInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-492], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005	

Table 439 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_007'.

**Table 439 — Test case description for
'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_007'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_007

Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_ServiceSelectionInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-492], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005	

Table 440 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_008'.

**Table 440 — Test case description for
'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_008'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_008
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_NoChargeServiceSelected' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-492], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005	

Table 441 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009'.

**Table 441 — Test case description for
'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009

Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryRes or ServiceDetailRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving the PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-492], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_006	

Table 442 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_010'.

**Table 442 — Test case description for
'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_010'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_010
Test objective	Test System executes GoodCase procedure and sends a valid AuthorizationRes message instead of PaymentServiceSelectionRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_007	

Table 443 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_011'.

**Table 443 — Test case description for
'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_011'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_011

Test objective	Test System executes GoodCase procedure, signals CP State E and sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_008	

Table 444 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_012'.

**Table 444 — Test case description for
'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_012'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_012
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_008	

10.3.2.4.2 AC specific test cases

Table 445 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001'.

**Table 445 — Test case description for
'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001'**

TC Id	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011	

Table 446 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002'.

**Table 446 — Test case description for
'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002'**

TC Id	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory

	parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012	

Table 447 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003'.

**Table 447 — Test case description for
'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003'**

TC Id	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380],

	[V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011	

Table 448 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004'.

**Table 448 — Test case description for
'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004'**

TC Id	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	

Expected behavior
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012

Table 449 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005'.

**Table 449 — Test case description for
'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005'**

TC Id	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.4.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011	

Table 450 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006'.

**Table 450 — Test case description for
'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006'**

TC Id	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID,

	ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012	

Table 451 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007'.

**Table 451 — Test case description for
'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007'**

TC Id	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory

	parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011	

Table 452 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008'.

**Table 452 — Test case description for
'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008'**

TC Id	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.2.1; 15118-2:8.4.2

Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012	

10.3.2.4.3 DC specific test cases

Table 453 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001'.

**Table 453 — Test case description for
'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001'**

TC Id	TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5 % duty cycle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause

PreCondition
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001
Expected behavior
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011

Table 454 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002'.

**Table 454 — Test case description for
'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002'**

TC Id	TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5 % duty cycle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.2.4; 15118-2:8.5.2.2.5; 15118-2:8.6.2.4.1; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
	f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001
Expected behavior	
	f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012

Table 455 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003'.

**Table 455 — Test case description for
'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003'**

TC Id	TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication

	session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.4.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dc, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011	

Table 456 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004'.

**Table 456 — Test case description for
'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004'**

TC Id	TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid

	ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.1; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012	

Table 457 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005'.

**Table 457 — Test case description for
'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005'**

TC Id	TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]

Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011	

Table 458 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006'.

**Table 458 — Test case description for
'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006'**

TC Id	TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.1; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode,	

f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001
Expected behavior
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012

Table 459 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007'.

**Table 459 — Test case description for
'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007'**

TC Id	TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011	

Table 460 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008'.

**Table 460 — Test case description for
'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008'**

TC Id	TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008

Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012	

10.3.2.5 EVCC test cases for PaymentServiceSelection

The following subclauses include all *V2G PaymentServiceSelection* message test case descriptions, where the EVCC is defined as SUT.

10.3.2.5.1 Common test cases

Table 461 lists the test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_001'.

Table 461 — Test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_001'

TC Id	TC_EVCC_CMN_VTB_PaymentServiceSelection_001
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq

	message with the current SessionID, SelectedPaymentOption 'Contract', valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.6; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-394], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_003	

Table 462 lists the test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_002'.

Table 462 — Test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_002'

TC Id	TC_EVCC_CMN_VTB_PaymentServiceSelection_002
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'ExternalPayment', valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.2.4.1; 15118-2:8.6.3.6; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-390], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_004	

Table 463 lists the test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_003'.

Table 463 — Test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_003'

TC Id	TC_EVCC_CMN_VTB_PaymentServiceSelection_003
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'Contract', valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_001	

Table 464 lists the test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_004'.

Table 464 — Test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_004'

TC Id	TC_EVCC_CMN_VTB_PaymentServiceSelection_004
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'ExternalPayment', valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	

Expected behavior
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_002

10.3.2.5.2 AC specific test cases

Table 465 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_001'.

Table 465 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_001'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_001
Test objective	Test System executes GoodCase procedure, initiates a oscillator shutdown, sets a nominal duty cycle of 53 % (32A) and sends a ServiceDiscoveryRes with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'ExternalPayment', valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.1; 15118-2:8.6.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-828]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PaymentServiceSelection_001	

Table 466 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_002'.

Table 466 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_002'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_002
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.2; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009	

Table 467 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_003'.

Table 467 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_003'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_003
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating an oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM,

	PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 468 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_004'.

Table 468 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_004'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_004
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003	

Table 469 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_005'.

Table 469 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_005'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_005
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004	

Table 470 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_006'.

Table 470 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_006'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_006
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.2; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009	

Table 471 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_007'.

Table 471 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_007'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_007
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM,

	PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 472 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_008'.

Table 472 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_008'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_008
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003	

Table 473 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_009'.

Table 473 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_009'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_009
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004	

Table 474 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_010'.

Table 474 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_010'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_010
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq

	message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.2; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009	

Table 475 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_011'.

Table 475 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_011'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_011
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-

	662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 476 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_012'.

Table 476 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_012'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_012
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	

Expected behavior
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003

Table 477 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_013'.

Table 477 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_013'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_013
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.4.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004	

Table 478 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_014'.

Table 478 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_014'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_014
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by

	the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.2; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009	

Table 479 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_015'.

Table 479 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_015'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_015
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_PmaxOW < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 480 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_016'.

Table 480 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_016'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_016
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_PmaxOW) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_PmaxOW' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName

	:= PKI_1, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003	

Table 481 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_017'.

Table 481 — Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_017'

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_017
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.3.1; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004	

10.3.2.5.3 DC specific test cases

Table 482 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_001'.

Table 482 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_001'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_001
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5 % duty cycle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.4.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.2; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009	

Table 483 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_002'.

Table 483 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_002'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_002
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5 % duty cycle and waits for successful link detection (old logical network

	parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.1; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 484 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_003'.

Table 484 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_003'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_003
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5 % duty cycle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-

requirement(s)	490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003	

Table 485 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_004'.

Table 485 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_004'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_004
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5 % duty cycle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.1; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	

f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004

Table 486 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_005'.

Table 486 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_005'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_005
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.2; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009	

Table 487 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_006'.

Table 487 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_006'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_006
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection

	(old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.1; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 488 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_007'.

Table 488 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_007'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_007
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-

requirement(s)	490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003	

Table 489 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_008'.

Table 489 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_008'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_008
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.1; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	

f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004

Table 490 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_009'.

Table 490 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_009'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_009
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.2; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009	

Table 491 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_010'.

Table 491 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_010'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_010
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within

	ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.1; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 492 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_011'.

Table 492 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_011'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_011
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003	

Table 493 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_012'.

Table 493 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_012'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_012
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'OK', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.1; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge,

	par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004	

Table 494 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_013'.

Table 494 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_013'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_013
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.2; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009	

Table 495 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_014'.

Table 495 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_014'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_014
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.4.4.2.1 15118-2:8.4.3.5.3; 15118-2:8.4.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.1; 15118-2:8.6.3.6; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 496 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_015'.

Table 496 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_015'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_015
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After

	'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003	

Table 497 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_016'.

Table 497 — Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_016'

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_016
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract' and 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.4.1; 15118-2:8.4.3.4.2; 15118-2:8.4.3.5.2; 15118-2:8.8.4.2.1; 15118-2:8.4.3.5.3; 15118-2:8.8.4.2.1; 15118-2:8.5.2.24; 15118-2:8.5.2.25; 15118-2:8.6.2.4.1; 15118-2:8.6.2.1; 15118-2:8.4.2

Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_PmaxOW < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004	

10.3.2.6 EVCC test cases for CertificateInstallation

The following subclauses include all *V2G CertificateInstallation message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.6.1 Common test cases

Table 498 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_001'.

Table 498 — Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_001'

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_001
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct CertificateInstallationReq message with a valid signature, current SessionID, OEMProvisioning certificate and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.8.4.2.1; 15118-2:8.4.2; 15118-2:8.4.3.11.2; 15118-2:8.5.2.27; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.3; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.5.2.26; 15118-2:7.3.3; 15118-2:7.9.2.4.3; 15118-2:AnnexF
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-235], [V2G2-236], [V2G2-496], [V2G2-747], [V2G2-355], [V2G2-356], [V2G2-386], [V2G2-398], [V2G2-411], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-893], [V2G2-909], [V2G2-353], [V2G2-354], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-926], [V2G2-822], [V2G2-932], [V2G2-933]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	

f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001
Expected behavior
f_EVCC_CMN_TB_VTB_CertificateInstallation_001

Table 499 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_002'.

Table 499 — Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_002'

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateInstallationRes message with current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_002	

Table 500 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_003'.

Table 500 — Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_003'

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateInstallationRes message with current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	

f_EVCC_CMN_TB_VTB_CertificateInstallation_002

Table 501 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_004'.

Table 501 — Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_004'

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateInstallationRes message with current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_002	

Table 502 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_005'.

Table 502 — Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_005'

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateInstallationRes message with current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_002	

Table 503 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_006'.

Table 503 — Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_006'

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateInstallationRes message with current SessionID, ResponseCode 'FAILED_CertificateExpired' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_002	

Table 504 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_007'.

Table 504 — Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_007'

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_007
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateInstallationRes message with current SessionID, ResponseCode 'FAILED_CertificateRevoked' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_002	

Table 505 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_008'.

Table 505 — Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_008'

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_008
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateInstallationRes message with current SessionID, ResponseCode 'FAILED_NoCertificateAvailable' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_002	

Table 506 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_009'.

Table 506 — Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_009'

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_009
Test objective	Test System executes GoodCase procedure, sends a valid PaymentServiceSelectionRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a CertificateInstallationReq message with a valid signature, current SessionID, OEMProvisioning certificate and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_003	

Table 507 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_010'.

Table 507 — Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_010'

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_010

Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of CertificateInstallationRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_004	

Table 508 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_011'.

Table 508 — Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_011'

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_011
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a CertificateInstallationRes message with a valid signature, current SessionID, ResponseCode 'OK', SAProvisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_005	

Table 509 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_012'.

Table 509 — Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_012'

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_012
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a CertificateInstallationRes message with a valid signature, current SessionID,

	ResponseCode 'OK', SAProvisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_005	

10.3.2.7 EVCC test cases for CertificateUpdate

The following subclauses include all *V2G CertificateUpdate* message test case descriptions, where the EVCC is defined as SUT.

10.3.2.7.1 Common test cases

Table 510 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_001'.

Table 510 — Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_001'

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_001
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct CertificateUpdateReq message with a valid signature, current SessionID, contract certificate chain, valid EMAID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.8.4.2.1; 15118-2:8.4.2; 15118-2:8.4.3.10.1; 15118-2:8.4.3.10.2; 15118-2:8.5.2.5; 15118-2:8.5.2.27; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.4; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.5.2.26; 15118-2:7.3.3; 15118-2:AnnexF; 15118-2:7.9.2.4.3; 15118-2:7.9.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-227], [V2G2-228], [V2G2-229], [V2G2-274], [V2G2-275], [V2G2-497], [V2G2-747], [V2G2-355], [V2G2-356], [V2G2-388], [V2G2-400], [V2G2-413], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-888], [V2G2-909], [V2G2-353], [V2G2-354], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-926], [V2G2-884], [V2G2-822], [V2G2-103], [V2G2-104]
Config Id	CF_04_002

PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_001	

Table 511 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_002'.

Table 511 — Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_002'

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 512 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_003'.

Table 512 — Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_003'

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName :=

	PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 513 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_004'.

Table 513 — Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_004'

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 514 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_005'.

Table 514 — Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_005'

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1

PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 515 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_006'.

Table 515 — Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_006'

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_CertificateExpired' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 516 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_007'.

Table 516 — Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_007'

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_007
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_CertificateRevoked' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate,

	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 517 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_008'.

Table 517 — Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_008'

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_008
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_NoCertificateAvailable' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 518 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_009'.

Table 518 — Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_009'

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_009
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_ContractCanceled' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1

PreCondition
f_EVCC_CMN_PR_CertificateUpdate_001
Expected behavior
f_EVCC_CMN_TB_VTB_CertificateUpdate_002

Table 519 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_010'.

Table 519 — Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_010'

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_010
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_ChallengeInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 520 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_011'.

Table 520 — Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_011'

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_011
Test objective	Test System executes GoodCase procedure, sends a valid PaymentServiceSelectionRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a CertificateUpdateReq message with a valid signature, current SessionID, valid contract certificate chain and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1

PreCondition
f_EVCC_CMN_PR_CertificateUpdate_001
Expected behavior
f_EVCC_CMN_TB_VTB_CertificateUpdate_003

Table 521 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_012'.

Table 521 — Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_012'

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_012
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of CertificateUpdateRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnc, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_004	

Table 522 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_013'.

Table 522 — Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_013'

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_013
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a CertificateUpdateRes message with a valid signature, current SessionID, ResponseCode 'OK', Saprovisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnc, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1

PreCondition
f_EVCC_CMN_PR_CertificateUpdate_001
Expected behavior
f_EVCC_CMN_TB_VTB_CertificateUpdate_005

Table 523 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_014'.

Table 523 — Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_014'

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_014
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a CertificateUpdateRes message with a valid signature, current SessionID, ResponseCode 'OK', SAProvisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_005	

10.3.2.8 EVCC test cases for CertificateHandlingAndPaymentDetails

The following subclauses include all V2G *CertificateHandlingAndPaymentDetails* message test case descriptions, where the EVCC is defined as SUT.

10.3.2.8.1 Common test cases

Table 524 lists the test case description for:

'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001'.

Table 524 — Test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001'

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionRes with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters. If a valid CertificateInstallationReq or CertificateUpdateReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.6.2; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1; 15118-2:8.5.2.26; 15118-2:7.3.2; 15118-2:7.3.3; 15118-2:AnnexF; 15118-2:7.9.2.4.3; 15118-2:7.9.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-205], [V2G2-206], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-898], [V2G2-353], [V2G2-354], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-926]; [V2G2-884], [V2G2-822], [V2G2-103], [V2G2-104]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := unknown, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_001	

Table 525 lists the test case description for:

'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_002'

**Table 525 — Test case description for
'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_002'**

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentDetailsRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-502], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_002	

Table 526 lists the test case description for:

'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_003'.

**Table 526 — Test case description for
'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_003'**

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_003

Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentDetailsRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-502], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_002	

Table 527 lists the test case description for:

'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_004'.

**Table 527 — Test case description for
'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_004'**

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentDetailsRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-502], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_002	

Table 528 lists the test case description for:

'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005'.

**Table 528 — Test case description for
'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005'**

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes,

	CertificateInstallationRes or CertificateUpdateRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-502], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_003	

Table 529 lists the test case description for:

'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_006'.

**Table 529 — Test case description for
'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_006'**

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_006
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of PaymentDetailsRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_004	

Table 530 lists the test case description for:

'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_007'.

**Table 530 — Test case description for
'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_007'**

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_007

Test objective	Test System executes GoodCase procedure, signals CP State E and sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_005	

Table 531 lists the test case description for:

'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_008'.

**Table 531 — Test case description for
'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_008'**

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_008
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_005	

10.3.2.9 EVCC test cases for PaymentDetails

The following subclauses include all *V2G PaymentDetails* message test case descriptions, where the EVCC is defined as SUT.

10.3.2.9.1 Common test cases

Table 532 lists the test case description for 'TC_EVCC_CMN_VTB_PaymentDetails_001'.

Table 532 — Test case description for 'TC_EVCC_CMN_VTB_PaymentDetails_001'

TC Id	TC_EVCC_CMN_VTB_PaymentDetails_001
Test objective	Test System executes GoodCase procedure and sends a CertificateInstallationRes message with a valid signature, current SessionID, ResponseCode 'OK', SAProvisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.6.2; 15118-2:8.4.4.2; 15118-2:8.4.4.2; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.3; 15118-2:7.9.2.4.5; 15118-2:8.5.2.26; 15118-2:7.3.3; 15118-2:AnnexF; 15118-2:8.4.3.11.3; 15118-2:7.9.2.4.3; 15118-2:7.9.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-205], [V2G2-206], [V2G2-500], [V2G2-747], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-108], [V2G2-652], [V2G2-653], [V2G2-898], [V2G2-909], [V2G2-353], [V2G2-354], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-926], [V2G2-884], [V2G2-896], [V2G2-822], [V2G2-103], [V2G2-104]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentDetails_001	

Table 533 lists the test case description for 'TC_EVCC_CMN_VTB_PaymentDetails_002'.

Table 533 — Test case description for 'TC_EVCC_CMN_VTB_PaymentDetails_002'

TC Id	TC_EVCC_CMN_VTB_PaymentDetails_002
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateRes message with a valid signature, current SessionID, ResponseCode 'OK', SAProvisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.6.2; 15118-2:8.4.4.2; 15118-2:8.4.4.2; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.3; 15118-2:7.9.2.4.5; 15118-2:8.5.2.26; 15118-2:7.3.3; 15118-2:8.4.3.10.3; 15118-2:7.9.2.4.3; 15118-2:7.9.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-205], [V2G2-206], [V2G2-501], [V2G2-747], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-108], [V2G2-652], [V2G2-653], [V2G2-898], [V2G2-909], [V2G2-353], [V2G2-354], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-926], [V2G2-891], [V2G2-822], [V2G2-103], [V2G2-104]
Config Id	CF_04_002

PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentDetails_002	

Table 534 lists the test case description for 'TC_EVCC_CMN_VTB_PaymentDetails_003'.

Table 534 — Test case description for 'TC_EVCC_CMN_VTB_PaymentDetails_003'

TC Id	TC_EVCC_CMN_VTB_PaymentDetails_003
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.6.2; 15118-2:8.8.4.2; 15118-2:8.4.2; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1; 15118-2:8.5.2.26; 15118-2:7.3.2; 15118-2:7.3.3; 15118-2:7.9.2.4.3; 15118-2:7.9.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-205], [V2G2-206], [V2G2-495], [V2G2-747], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-898], [V2G2-353], [V2G2-354], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-926], [V2G2-822], [V2G2-103], [V2G2-104]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentDetails_003	

10.3.2.10 EVCC test cases for Authorization

The following subclauses include all *V2G Authorization message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.10.1 Common test cases

Table 535 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_001'.

Table 535 — Test case description for 'TC_EVCC_CMN_VTB_Authorization_001'

TC Id	TC_EVCC_CMN_VTB_Authorization_001
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends a correct AuthorizationReq message with a

	valid signature, the current SessionID, valid GenChallenge and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.1; 15118-2:8.7.4.2; 15118-2:8.8.4.2.1; 15118-2:8.8.4.2.2; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-210], [V2G2-211], [V2G2-747], [V2G2-503], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-900], [V2G2-909]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_001	

Table 536 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_002'.

Table 536 — Test case description for 'TC_EVCC_CMN_VTB_Authorization_002'

TC Id	TC_EVCC_CMN_VTB_Authorization_002
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct AuthorizationReq message with the current SessionID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.1; 15118-2:8.7.4.2; 15118-2:8.8.4.2.1; 15118-2:8.8.4.2.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.4.1; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-210], [V2G2-211], [V2G2-747], [V2G2-509], [V2G2-376], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_002	

Table 537 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_003'.

Table 537 — Test case description for 'TC_EVCC_CMN_VTB_Authorization_003'

TC Id	TC_EVCC_CMN_VTB_Authorization_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending an AuthorizationRes message with the current SessionID, ResponseCode 'FAILED' and

	all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-504], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_003	

Table 538 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_004'.

Table 538 — Test case description for 'TC_EVCC_CMN_VTB_Authorization_004'

TC Id	TC_EVCC_CMN_VTB_Authorization_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending an AuthorizationRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-504], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_003	

Table 539 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_005'.

Table 539 — Test case description for 'TC_EVCC_CMN_VTB_Authorization_005'

TC Id	TC_EVCC_CMN_VTB_Authorization_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending an AuthorizationRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-504], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_003	

Table 540 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_006'.

Table 540 — Test case description for 'TC_EVCC_CMN_VTB_Authorization_006'

TC Id	TC_EVCC_CMN_VTB_Authorization_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending an AuthorizationRes message with the current SessionID, ResponseCode 'FAILED_ChallengeInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-504], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_003	

Table 541 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_007'.

Table 541 — Test case description for 'TC_EVCC_CMN_VTB_Authorization_007'

TC Id	TC_EVCC_CMN_VTB_Authorization_007
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending an AuthorizationRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-504], [V2G2-728], [V2G2-025]

Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_PkiName := PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_003	

Table 542 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_008'.

Table 542 — Test case description for 'TC_EVCC_CMN_VTB_Authorization_008'

TC Id	TC_EVCC_CMN_VTB_Authorization_008
Test objective	Test System executes GoodCase procedure and sends a valid ServicePaymentSelectionRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a AuthorizationReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-504], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_004	

Table 543 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_009'.

Table 543 — Test case description for 'TC_EVCC_CMN_VTB_Authorization_009'

TC Id	TC_EVCC_CMN_VTB_Authorization_009
Test objective	Test System executes GoodCase procedure, sends continuously an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Ongoing' and all additional mandatory parameters and waits for another AuthorizationReq message with the current SessionID and all additional mandatory parameters until the V2G_EVCC_Ongoing_Timer is equal or larger than V2G_EVCC_Ongoing_Timeout. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.3.1; 15118-2:8.7.3.4; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-605], [V2G2-710], [V2G2-711], [V2G2-684], [V2G2-845]

Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_005	

Table 544 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_010'.

Table 544 — Test case description for 'TC_EVCC_CMN_VTB_Authorization_010'

TC Id	TC_EVCC_CMN_VTB_Authorization_010
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends a correct AuthorizationReq message with a valid signature, the current SessionID, valid GenChallenge and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.1; 15118-2:8.7.4.2; 15118-2:8.8.4.2.1; 15118-2:8.8.4.2.2; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:7.9.2.4.3; 15118-2:8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-210], [V2G2-211], [V2G2-747], [V2G2-503], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-900], [V2G2-909], [V2G2-121], [V2G2-122], [V2G2-817], [V2G2-818], [V2G2-896]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling:= certInstall, PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_001	

Table 545 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_011'.

Table 545 — Test case description for 'TC_EVCC_CMN_VTB_Authorization_011'

TC Id	TC_EVCC_CMN_VTB_Authorization_011
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends a correct AuthorizationReq message with a valid signature, the current SessionID, valid GenChallenge and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.7.1; 15118-2:8.7.4.2; 15118-2:8.8.4.2.1; 15118-2:8.8.4.2.2; 15118-2:8.6.2.3.2; 15118-2:8.6.2.4.2;

	15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:7.9.2.4.3; 15118-2:8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-210], [V2G2-211], [V2G2-747], [V2G2-503], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-900], [V2G2-909], [V2G2-121], [V2G2-122], [V2G2-230], [V2G2-817], [V2G2-818], [V2G2-827], [V2G2-891]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling:= certUpdate, PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_001	

Table 546 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_012'.

Table 546 — Test case description for 'TC_EVCC_CMN_VTB_Authorization_012'

TC Id	TC_EVCC_CMN_VTB_Authorization_012
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of AuthorizationRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_006	

Table 547 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_013'.

Table 547 — Test case description for 'TC_EVCC_CMN_VTB_Authorization_013'

TC Id	TC_EVCC_CMN_VTB_Authorization_013
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_007	

Table 548 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_014'.

Table 548 — Test case description for 'TC_EVCC_CMN_VTB_Authorization_014'

TC Id	TC_EVCC_CMN_VTB_Authorization_014
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_007	

10.3.2.11 EVCC test cases for ChargeParameterDiscovery

The following subclauses include all *V2G ChargeParameterDiscovery message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.11.1 Common test cases

Table 549 lists the test case description for 'TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001'.

Table 549 — Test case description for 'TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001'

TC Id	TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a valid AuthorizationRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters.

	Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_002	

Table 550 lists the test case description for 'TC_EVCC_CMN_VTB_ChargeParameterDiscovery_002'.

Table 550 — Test case description for 'TC_EVCC_CMN_VTB_ChargeParameterDiscovery_002'

TC Id	TC_EVCC_CMN_VTB_ChargeParameterDiscovery_002
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of ChargeParameterDiscoveryRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_003	

10.3.2.11.2 AC specific test cases

Table 551 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_001'.

Table 551 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_001'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType and all

	additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= unknown
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_001	

Table 552 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_002'.

Table 552 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_002'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 553 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_003'.

Table 553 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_003'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 554 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_004'.

Table 554 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_004'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 555 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_005'.

Table 555 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_005'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongEnergyTransferType' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]

Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 556 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_006'.

Table 556 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_006'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongChargeParameter' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 557 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_007'.

Table 557 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_007'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_007
Test objective	Test System executes GoodCase procedure, sends continuously a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Ongoing', a valid scheduleList and all additional mandatory parameters and waits for another valid ChargeParameterDiscoveryReq message until the V2G_EVCC_Ongoing_Timer is equal or larger than V2G_EVCC_Ongoing_Timeout. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.3.1; 15118-2:8.7.3.4
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-605], [V2G2-710], [V2G2-711], [V2G2-685]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC

PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_002	

Table 558 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_008'.

Table 558 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_008'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_008
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_EVCC_CMN_DepartureTime:= unknown
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_003	

Table 559 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_009'.

Table 559 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_009'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_009
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]

Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_004	

Table 560 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_010'.

Table 560 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_010'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_010
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_004	

Table 561 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_011'.

Table 561 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_011'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_011
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376],

	[V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_CMN_DepartureTime:= unknown
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_003	

Table 562 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_012'.

Table 562 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_012'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_012
Test objective	Test System executes GoodCase procedure and sends a AuthorizationRes message with with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= none_
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 563 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_013'.

Table 563 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_013'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_013
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS

	Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-215]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_EVCC_CMN_DepartureTime:= none_
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 564 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_014'.

Table 564 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_014'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_014
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-215]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_CMN_DepartureTime:= none_
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 565 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_015'.

Table 565 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_015'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_015

Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 566 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_016'.

Table 566 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_016'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_016
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1;

	15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= none_, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005	

Table 567 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_017'.

Table 567 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_017'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_017
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= departureTime, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	

f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004

Table 568 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_018'.

Table 568 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_018'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_018
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= unknown, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 569 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_019'.

Table 569 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_019'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_019
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters.

	Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= none_, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005	

Table 570 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_020'.

Table 570 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_020'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_020
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= departureTime,

	PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 571 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_021'.

Table 571 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_021'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_021
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= unknown
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 572 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_022'.

Table 572 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_022'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_022
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID,

	ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.4.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= none_
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_SetPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005	

Table 573 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_023'.

Table 573 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_023'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_023
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-

	2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= departureTime
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 574 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_024'.

Table 574 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_024'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_024
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= unknown
PreCondition	

f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001
Expected behavior
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004

Table 575 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_025'.

Table 575 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_025'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_025
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= none_
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005	

Table 576 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_026'.

Table 576 — Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_026'

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_026
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid

	SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.2.7; 15118-2:8.5.3.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.2.1; 15118-2:8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= departureTime
PreCondition	
f_EVCC_AC_PR_SessionStop_002, f_EVCC_SetPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

10.3.2.11.3 DC specific test cases

Table 577 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_001'.

Table 577 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_001'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC

PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= unknown
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_001	

Table 578 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_002'.

Table 578 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_002'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dc
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 579 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_003'.

Table 579 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_003'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dc
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	

Expected behavior
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001

Table 580 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_004'.

Table 580 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_004'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 581 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_005'.

Table 581 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_005'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongEnergyTransferType' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 582 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_006'.

Table 582 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_006'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongChargeParameter' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 583 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_007'.

Table 583 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_007'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_007
Test objective	Test System executes GoodCase procedure, sends continuously a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Ongoing', a valid scheduleList and all additional mandatory parameters and waits for another valid ChargeParameterDiscoveryReq message until the V2G_EVCC_Ongoing_Timer is equal or larger than V2G_EVCC_Ongoing_Timeout. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.3.1; 15118-2:8.7.3.4
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-605], [V2G2-710], [V2G2-711], [V2G2-685]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_002	

Table 584 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_008'.

Table 584 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_008'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_008
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all

	additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_Renegotiation:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_EVCC_CMN_DepartureTime:= unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_003	

Table 585 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_009'.

Table 585 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_009'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_009
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_004	

Table 586 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_010'.

Table 586 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_010'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_010
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_004	

Table 587 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_011'.

Table 587 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_011'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_011
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_Renegotiation:= false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_CMN_DepartureTime:= unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_003	

Table 588 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_012'.

Table 588 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_012'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_012
Test objective	Test System executes GoodCase procedure and sends a AuthorizationRes message with with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-215]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= none_
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 589 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_013'.

Table 589 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_013'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_013
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-215]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_Renegotiation:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown,

	PIXIT_EVCC_CMN_DepartureTime:= none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 590 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_014'.

Table 590 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_014'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_014
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-215]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_Renegotiation:= false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_CMN_DepartureTime:= none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 591 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_015'.

Table 591 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_015'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_015
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5 % duty cycle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and

	all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= unknown, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_SetPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 592 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_016'.

Table 592 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_016'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_016
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5% duty cycle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= none_, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	

f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001
Expected behavior
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005

Table 593 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_017'.

Table 593 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_017'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_017
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5 % duty cycle and waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.4.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= departureTime, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 594 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_018'.

Table 594 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_018'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_018
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase

	procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= unknown, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 595 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_019'.

Table 595 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_019'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_019
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002

PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= none_, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005	

Table 596 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_020'.

Table 596 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_020'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_020
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.4.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= departureTime, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 597 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_021'.

Table 597 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_021'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_021

Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= unknown
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 598 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_022'.

Table 598 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_022'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_022
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= none_
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005	

Table 599 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_023'.

Table 599 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_023'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_023
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true

PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= departureTime
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 600 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_024'.

Table 600 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_024'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_024
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= unknown
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 601 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_025'.

Table 601 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_025'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_025
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID,

	ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.8.2; 15118-2:8.4.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= none_
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005	

Table 602 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_026'.

Table 602 — Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_026'

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_026
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System waits for termination of the logical network by the SUT, still detecting CP State B within the entire sleep mode. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) triggered by the SUT. Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-

	2:8.4.3.8.2; 15118-2:8.8.4.2.1; 15118-2:8.5.4.3 15118-2:8.5.2.7; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting:= true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime:= departureTime
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

10.3.2.12 EVCC test cases for PowerDelivery

The following subclauses include all *V2G PowerDelivery* message test case descriptions, where the EVCC is defined as SUT.

10.3.2.12.1 AC specific test cases

Table 603 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_001'.

Table 603 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_001'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_001
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State C or D within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-846], [V2G2-510], [V2G2-847], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-293], [V2G2-673], [V2G2-846]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization:= unknown
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	

Expected behavior
f_EVCC_AC_TB_VTB_PowerDelivery_001

Table 604 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_002'.

Table 604 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_002'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_002	

Table 605 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_003'.

Table 605 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_003'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_002	

Table 606 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_004'.

Table 606 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_004'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_002	

Table 607 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_005'.

Table 607 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_005'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ChargingProfileInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_002	

Table 608 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_006'.

Table 608 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_006'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a

	PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_TariffSelectionInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_002	

Table 609 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_007'.

Table 609 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_007'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_007
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_PowerDeliveryNotApplied' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_002	

Table 610 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_008'.

Table 610 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_008'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_008
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ContactorError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within

	'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_002	

Table 611 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_009'.

Table 611 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_009'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_009
Test objective	Test System executes GoodCase procedure and sends a valid ChargeParameterDiscoveryRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_003	

Table 612 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_010'.

Table 612 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_010'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_010
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of PowerDeliveryRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1

Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_004	

Table 613 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_011'.

Table 613 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_011'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_011
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_005	

Table 614 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_012'.

Table 614 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_012'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_012
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002

PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_005	

Table 615 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_013'.

Table 615 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_013'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_013
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization:= unknown, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 616 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_014'.

Table 616 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_014'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_014
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS

	Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization:= unknown, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 617 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_015'.

617 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_015'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_015
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including MaxNumberOfPhasesInUse and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State C or D within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-846], [V2G2-510], [V2G2-847], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-829], [V2G2-293], [V2G2-673], [V2G2-846]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization:= unknown
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_001	

Table 618 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_016'.

Table 618 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_016'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_016
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including MaxNumberOfPhasesInUse and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization:= unknown, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 619 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_017'.

Table 619 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_017'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_017
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including MaxNumberOfPhasesInUse and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-675], [V2G2-673]

Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization:= unknown, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 620 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_018'.

Table 620 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_018'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_018
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State C or D within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-846], [V2G2-510], [V2G2-847], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-293], [V2G2-296], [V2G2-675], [V2G2-673], [V2G2-846]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_EVCC_SalesTariff:= true
PIXIT selection	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization:= optimization
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_001	

Table 621 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_019'.

Table 621 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_019'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_019
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a

	valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including MaxNumberOfPhasesInUse and the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State C or D within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-846], [V2G2-510], [V2G2-847], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-829], [V2G2-293], [V2G2-296], [V2G2-675], [V2G2-673], [V2G2-846]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_EVCC_SalesTariff:= true
PIXIT selection	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization:= optimization
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_001	

Table 622 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_020'.

Table 622 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_020'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_020
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including the default SAScheduleTuple and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State C or D within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-846], [V2G2-510], [V2G2-847], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-675], [V2G2-673], [V2G2-846]
Config Id	CF_04_002

PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_EVCC_SalesTariff:= true
PIXIT selection	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization:= none_
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_001	

Table 623 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_021'.

Table 623 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_021'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_021
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including MaxNumberOfPhasesInUse and the default SAScheduleTuple and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State C or D within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-846], [V2G2-510], [V2G2-847], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-829], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-675], [V2G2-673], [V2G2-846]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_EVCC_SalesTariff:= true
PIXIT selection	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization:= none_
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_001	

Table 624 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_022'.

Table 624 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_022'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_022
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs and all additional mandatory parameters.

	Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-296], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= true, PICS_CMN_EVCC_SalesTariff:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization:= optimization, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 625 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_023'.

Table 625 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_023'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_023
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-296], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= false, PICS_CMN_EVCC_SalesTariff:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= unknown,

	PIXIT_EVCC_CMN_ChargingProfileOptimization:= optimization, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 626 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_024'.

Table 626 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_024'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_024
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including MaxNumberOfPhasesInUse and the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-296], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= true, PICS_CMN_EVCC_SalesTariff:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization:= optimization, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 627 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_025'.

Table 627 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_025'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_025
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including MaxNumberOfPhasesInUse and the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-296], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= false, PICS_CMN_EVCC_SalesTariff:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization:= optimization, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 628 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_026'.

Table 628 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_026'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_026
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including the default SAScheduleTuple and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= true, PICS_CMN_EVCC_SalesTariff:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization:= none_, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	

f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001
Expected behavior
f_EVCC_AC_TB_VTB_PowerDelivery_006

Table 629 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_027'.

Table 629 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_027'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_027
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including the default SAScheduleTuple and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= false, PICS_CMN_EVCC_SalesTariff:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization:= none_, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 630 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_028'.

Table 630 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_028'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_028
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including MaxNumberOfPhasesInUse and the default SAScheduleTuple and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2;

	15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= true, PICS_CMN_EVCC_SalesTariff:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization:= none_, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 631 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_029'.

Table 631 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_029'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_029
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including MaxNumberOfPhasesInUse and the default SAScheduleTuple and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= false, PICS_CMN_EVCC_SalesTariff:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization:= none_, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	

f_EVCC_AC_TB_VTB_PowerDelivery_006

Table 632 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_030'.

Table 632 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_030'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_030
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', the previous charging profile including the same SAScheduleTupleID than before renegotiation and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization:= unknown, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= newSAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 633 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_031'.

Table 633 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_031'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_031
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', the previous charging profile including the same SAScheduleTupleID than before renegotiation and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.8.4.1

Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization:= unknown, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= newSAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 634 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_032'.

Table 634 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_032'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_032
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', the previous charging profile including MaxNumberOfPhasesInUse and the same SAScheduleTupleID than before renegotiation and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization:= unknown, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= newSAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 635 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_033'.

Table 635 — Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_033'

TC Id	TC_EVCC_AC_VTB_PowerDelivery_033
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', the previous charging profile including MaxNumberOfPhasesInUse and the same SAScheduleTupleID than before renegotiation and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2; 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.3; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_Renegotiation:= false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse:= maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization:= unknown, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= newSAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

10.3.2.13 EVCC test cases for SessionStop**10.3.2.13.1 General**

The following subclauses include all *V2G SessionStop message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.13.2 AC specific test cases

Table 636 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_001'.

Table 636 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_001'

TC Id	TC_EVCC_AC_VTB_SessionStop_001
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with

	the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-508], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-660]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	PIXIT_EVCC_CMN_TLS:= tls, PIXIT_EVCC_CMN_Pause:= unknown
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 637 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_002'.

Table 637 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_002'

TC Id	TC_EVCC_AC_VTB_SessionStop_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionStopRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_SessionStop_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_003	

Table 638 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_003'.

Table 638 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_003'

TC Id	TC_EVCC_AC_VTB_SessionStop_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionStopRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within

	'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_SessionStop_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_003	

Table 639 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_004'.

Table 639 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_004'

TC Id	TC_EVCC_AC_VTB_SessionStop_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionStopRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_SessionStop_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_003	

Table 640 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_005'.

Table 640 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_005'

TC Id	TC_EVCC_AC_VTB_SessionStop_005
Test objective	Test System executes GoodCase procedure, sends a valid PowerDeliveryRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a SessionStopReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2

Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_SessionStop_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_004	

Table 641 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_006'.

Table 641 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_006'

TC Id	TC_EVCC_AC_VTB_SessionStop_006
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of SessionStopRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_SessionStop_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_005	

Table 642 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_007'.

Table 642 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_007'

TC Id	TC_EVCC_AC_VTB_SessionStop_007
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.4.2.2; 15118-2:8.6.2.3.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1;

	15118-2:8.6.2.1; 15118-2:7.3.4
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-633], [V2G2-635], [V2G2-637]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS:= none_, PIXIT_EVCC_CMN_Pause:= unknown
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 643 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_008'.

Table 643 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_008'

TC Id	TC_EVCC_AC_VTB_SessionStop_008
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.4.2.2; 15118-2:8.6.2.3.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS:= unknown, PIXIT_EVCC_CMN_Pause:= unknown
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 644 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_009'.

Table 644 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_009'

TC Id	TC_EVCC_AC_VTB_SessionStop_009
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and

	all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:7.4
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	PIXIT_EVCC_CMN_TLS:= tls, PIXIT_EVCC_CMN_Pause:= none_
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 645 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_010'.

Table 645 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_010'

TC Id	TC_EVCC_AC_VTB_SessionStop_010
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.4.2.2; 15118-2:8.6.2.3.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:7.3.4; 15118-2:7.4
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS:= none_, PIXIT_EVCC_CMN_Pause:= none_
PreCondition	

f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001
Expected behavior
f_EVCC_AC_TB_VTB_SessionStop_001

Table 646 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_011'.

Table 646 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_011'

TC Id	TC_EVCC_AC_VTB_SessionStop_011
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.4.2.2; 15118-2:8.6.2.3.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:7.4
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS:= unknown, PIXIT_EVCC_CMN_Pause:= none_
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 647 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_012'.

Table 647 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_012'

TC Id	TC_EVCC_AC_VTB_SessionStop_012
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-

	2:8.4.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	PIXIT_EVCC_CMN_TLS:= tls, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 648 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_013'.

Table 648 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_013'

TC Id	TC_EVCC_AC_VTB_SessionStop_013
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.4.2.2; 15118-2:8.6.2.3.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:7.3.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS:= none_, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 649 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_014'.

Table 649 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_014'

TC Id	TC_EVCC_AC_VTB_SessionStop_014
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message

	with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.4.2.2; 15118-2:8.6.2.3.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS:= unknown, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 650 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_015'.

Table 650 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_015'

TC Id	TC_EVCC_AC_VTB_SessionStop_015
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge

PIXIT selection	PIXIT_EVCC_CMN_TLS:= tls
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 651 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_016'.

Table 651 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_016'

TC Id	TC_EVCC_AC_VTB_SessionStop_016
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.4.2.2; 15118-2:8.6.2.3.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:7.3.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_IdentificationMode:= eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_CMN_TLS:= none_
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 652 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_017'.

Table 652 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_017'

TC Id	TC_EVCC_AC_VTB_SessionStop_017
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all

	additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.4.2.2; 15118-2:8.6.2.3.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_IdentificationMode:= eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_CMN_TLS:= unknown
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 653 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_018'.

Table 653 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_018'

TC Id	TC_EVCC_AC_VTB_SessionStop_018
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge
PIXIT selection	PIXIT_EVCC_CMN_TLS:= tls

PreCondition
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001
Expected behavior
f_EVCC_AC_TB_VTB_SessionStop_001

Table 654 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_019'.

Table 654 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_019'

TC Id	TC_EVCC_AC_VTB_SessionStop_019
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.4.2.2; 15118-2:8.6.2.3.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:7.3.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_IdentificationMode:= eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge
PIXIT selection	PIXIT_EVCC_CMN_TLS:= none_
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 655 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_020'.

Table 655 — Test case description for 'TC_EVCC_AC_VTB_SessionStop_020'

TC Id	TC_EVCC_AC_VTB_SessionStop_020
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT

	terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.4.2.2; 15118-2:8.6.2.3.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_IdentificationMode:= eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge
PIXIT selection	PIXIT_EVCC_CMN_TLS:= unknown
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

10.3.2.13.3 DC specific test cases

Table 656 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_001'.

Table 656 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_001'

TC Id	TC_EVCC_DC_VTB_SessionStop_001
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.12; 15118-2:8.4.5.5; 15118-2:8.4.1; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection, PIXIT_EVCC_CMN_TLS:= tls, PIXIT_EVCC_CMN_Pause:= unknown
PreCondition	

f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001
Expected behavior
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002

Table 657 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_002'.

Table 657 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_002'

TC Id	TC_EVCC_DC_VTB_SessionStop_002
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.12; 15118-2:8.4.5.5; 15118-2:8.4.1; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dc
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= none, PIXIT_EVCC_CMN_TLS:= tls, PIXIT_EVCC_CMN_Pause:= unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 658 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_003'.

Table 658 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_003'

TC Id	TC_EVCC_DC_VTB_SessionStop_003
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:7.3.4; 15118-

	2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection, PIXIT_EVCC_CMN_TLS:= none_, PIXIT_EVCC_CMN_Pause:= unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 659 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_004'.

Table 659 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_004'

TC Id	TC_EVCC_DC_VTB_SessionStop_004
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:7.3.4; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= none_, PIXIT_EVCC_CMN_TLS:= none_, PIXIT_EVCC_CMN_Pause:= unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 660 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_005'.

Table 660 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_005'

TC Id	TC_EVCC_DC_VTB_SessionStop_005

Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection, PIXIT_EVCC_CMN_TLS:= unknown, PIXIT_EVCC_CMN_Pause:= unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 661 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_006'.

Table 661 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_006'

TC Id	TC_EVCC_DC_VTB_SessionStop_006
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM

PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= none_, PIXIT_EVCC_CMN_TLS:= unknown, PIXIT_EVCC_CMN_Pause:= unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 662 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_007'.

Table 662 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_007'

TC Id	TC_EVCC_DC_VTB_SessionStop_007
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.12; 15118-2:8.4.5.5; 15118-2:8.4.1; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection, PIXIT_EVCC_CMN_TLS:= tls, PIXIT_EVCC_CMN_Pause:= none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 663 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_008'.

Table 663 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_008'

TC Id	TC_EVCC_DC_VTB_SessionStop_008
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with

	the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.12; 15118-2:8.4.5.5; 15118-2:8.4.1; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= none_, PIXIT_EVCC_CMN_TLS:= tls, PIXIT_EVCC_CMN_Pause:= none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 664 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_009'.

Table 664 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_009'

TC Id	TC_EVCC_DC_VTB_SessionStop_009
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:7.3.4; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection, PIXIT_EVCC_CMN_TLS:= none_, PIXIT_EVCC_CMN_Pause:= none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 665 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_010'.

Table 665 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_010'

TC Id	TC_EVCC_DC_VTB_SessionStop_010
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:7.3.4; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= none_, PIXIT_EVCC_CMN_TLS:= none_, PIXIT_EVCC_CMN_Pause:= none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 666 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_011'.

Table 666 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_011'

TC Id	TC_EVCC_DC_VTB_SessionStop_011
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4

Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection, PIXIT_EVCC_CMN_TLS:= unknown, PIXIT_EVCC_CMN_Pause:= none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 667 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_012'.

Table 667 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_012'

TC Id	TC_EVCC_DC_VTB_SessionStop_012
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= none_, PIXIT_EVCC_CMN_TLS:= unknown, PIXIT_EVCC_CMN_Pause:= none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 668 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_013'.

Table 668 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_013'

TC Id	TC_EVCC_DC_VTB_SessionStop_013
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory

	parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.12; 15118-2:8.4.5.5; 15118-2:8.4.1; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection, PIXIT_EVCC_CMN_TLS:= tls, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 669 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_014'.

Table 669 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_014'

TC Id	TC_EVCC_DC_VTB_SessionStop_014
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.12; 15118-2:8.4.5.5; 15118-2:8.4.1; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002

PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= none_, PIXIT_EVCC_CMN_TLS:= tls, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 670 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_015'.

Table 670 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_015'

TC Id	TC_EVCC_DC_VTB_SessionStop_015
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:7.3.4; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection, PIXIT_EVCC_CMN_TLS:= none_, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 671 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_016'.

Table 671 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_016'

TC Id	TC_EVCC_DC_VTB_SessionStop_016
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all

	additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:7.3.4; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= none_, PIXIT_EVCC_CMN_TLS:= none_, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 672 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_017'.

Table 672 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_017'

TC Id	TC_EVCC_DC_VTB_SessionStop_017
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection, PIXIT_EVCC_CMN_TLS:= unknown, PIXIT_EVCC_CMN_Pause:= pause

PreCondition
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001
Expected behavior
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002

Table 673 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_018'.

Table 673 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_018'

TC Id	TC_EVCC_DC_VTB_SessionStop_018
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= none, PIXIT_EVCC_CMN_TLS:= unknown, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 674 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_019'.

Table 674 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_019'

TC Id	TC_EVCC_DC_VTB_SessionStop_019
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within

	'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.12; 15118-2:8.4.5.5; 15118-2:8.4.1; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection, PIXIT_EVCC_CMN_TLS:= tls
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 675 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_020'.

Table 675 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_020'

TC Id	TC_EVCC_DC_VTB_SessionStop_020
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.12; 15118-2:8.4.5.5; 15118-2:8.4.1; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= none_, PIXIT_EVCC_CMN_TLS:= tls

PreCondition
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001
Expected behavior
f_EVCC_DC_TB_VTB_SessionStop_001

Table 676 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_021'.

Table 676 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_021'

TC Id	TC_EVCC_DC_VTB_SessionStop_021
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:7.3.4; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection, PIXIT_EVCC_CMN_TLS:= none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 677 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_022'.

Table 677 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_022'

TC Id	TC_EVCC_DC_VTB_SessionStop_022
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all

	additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:7.3.4; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= none, PIXIT_EVCC_CMN_TLS:= none
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 678 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_023'.

Table 678 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_023'

TC Id	TC_EVCC_DC_VTB_SessionStop_023
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM,

	PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection, PIXIT_EVCC_CMN_TLS:= unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 679 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_024'.

Table 679 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_024'

TC Id	TC_EVCC_DC_VTB_SessionStop_024
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= none, PIXIT_EVCC_CMN_TLS:= unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 680 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_025'.

Table 680 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_025'

TC Id	TC_EVCC_DC_VTB_SessionStop_025
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT sends a correct SessionStopReq message with the

	current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.12; 15118-2:8.4.5.5; 15118-2:8.4.1; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection, PIXIT_EVCC_CMN_TLS:= tls
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 681 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_026'.

Table 681 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_026'

TC Id	TC_EVCC_DC_VTB_SessionStop_026
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.12; 15118-2:8.4.5.5; 15118-2:8.4.1; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002

PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= none_, PIXIT_EVCC_CMN_TLS:= tls
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 682 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_027'.

Table 682 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_027'

TC Id	TC_EVCC_DC_VTB_SessionStop_027
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection with 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:7.3.4; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection, PIXIT_EVCC_CMN_TLS:= none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 683 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_028'.

Table 683 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_028'

TC Id	TC_EVCC_DC_VTB_SessionStop_028
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with

	the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:7.3.4; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= none_, PIXIT_EVCC_CMN_TLS:= none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 684 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_029'.

Table 684 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_029'

TC Id	TC_EVCC_DC_VTB_SessionStop_029
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]

Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection, PIXIT_EVCC_CMN_TLS:= unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 685 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_030'.

Table 685 — Test case description for 'TC_EVCC_DC_VTB_SessionStop_030'

TC Id	TC_EVCC_DC_VTB_SessionStop_030
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.12; 15118-2:8.4.2; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.8.4.2.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= none_, PIXIT_EVCC_CMN_TLS:= unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

10.3.2.14 EVCC test cases for ChargingStatus

The following subclause includes all V2G *ChargingStatus message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.14.1 AC specific test cases

Table 686 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_001'.

Table 686 — Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_001'

TC Id	TC_EVCC_AC_VTB_ChargingStatus_001
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargingStatusReq message with the current SessionID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.4.2.2; 15118-2:8.4.4.2.3; 15118-2:8.8.4.2; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-242], [V2G2-514], [V2G2-747], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_001	

Table 687 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_002'.

Table 687 — Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_002'

TC Id	TC_EVCC_AC_VTB_ChargingStatus_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargingStatusRes message with the current sessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-511], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_002	

Table 688 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_003'.

Table 688 — Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_003'

TC Id	TC_EVCC_AC_VTB_ChargingStatus_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargingStatusRes message with the current sessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-511], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_002	

Table 689 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_004'.

Table 689 — Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_004'

TC Id	TC_EVCC_AC_VTB_ChargingStatus_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargingStatusRes message with the current sessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-511], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_002	

Table 690 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_005'.

Table 690 — Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_005'

TC Id	TC_EVCC_AC_VTB_ChargingStatus_005
Test objective	Test System executes GoodCase procedure and sends a valid PowerDeliveryRes

	message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-511], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_003	

Table 691 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_006'.

Table 691 — Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_006'

TC Id	TC_EVCC_AC_VTB_ChargingStatus_006
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_004	

Table 692 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_007'.

Table 692 — Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_007'

TC Id	TC_EVCC_AC_VTB_ChargingStatus_007
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as

	soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_004	

Table 693 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_008'.

Table 693 — Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_008'

TC Id	TC_EVCC_AC_VTB_ChargingStatus_008
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of ChargingStatusRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication Session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_005	

10.3.2.15 EVCC test cases for ChargingLoopAcOrPowerDelivery

The following subclause includes all *V2G ChargingLoopAcOrPowerDelivery message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.15.1 AC specific test cases

Table 694 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001'.

Table 694 — Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001'

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001
Test objective	Test System executes GoodCase procedure and sends a number (PICS_CMN_CMN_LoopCounter) of ChargingStatusRes messages with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. In the last sequence EVSENotification will be changed to 'StopCharging'.

	Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State B within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.4.2.2; 15118-2:8.4.4.2.3; 15118-2:8.8.4.1; 15118-2:8.8.4.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-839], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-513], [V2G2-848], [V2G2-242], [V2G2-516], [V2G2-679], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-849], [V2G2-832], [V2G2-293], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_EVCC_CMN_ReceiptRequired:= false
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_001	

Table 695 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002'.

Table 695 — Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002'

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002
Test objective	Test System executes GoodCase procedure and sends a number (PICS_CMN_CMN_LoopCounter) of ChargingStatusRes messages with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters, always followed by MeteringReceiptRes messages. In the last sequence EVSENotification will be changed to 'StopCharging'. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters after receiving a MeteringReceiptRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State B within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.4.2.2; 15118-2:8.4.4.2.3; 15118-2:8.8.4.1; 15118-2:8.8.4.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-839], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-848], [V2G2-242], [V2G2-519], [V2G2-679], [V2G2-380], [V2G2-404], [V2G2-518], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-849], [V2G2-832], [V2G2-903], [V2G2-293], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1

PreCondition
f_EVCC_AC_PR_ChargingStatus_001
Expected behavior
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_002

Table 696 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_003'.

Table 696 — Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_003'

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003	

Table 697 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_004'.

Table 697 — Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_004'

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	

f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003

Table 698 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_005'.

Table 698 — Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_005'

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003	

Table 699 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_006'.

Table 699 — Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_006'

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ContactorError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003	

Table 700 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007'.

Table 700 — Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007'

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007
Test objective	Test System executes GoodCase procedure, sends a valid ChargingStatusRes or MeteringReceiptRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_004	

Table 701 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008'.

Table 701 — Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008'

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008
Test objective	Test System executes GoodCase procedure and initiates a renegotiation by sending a ChargingStatusRes message with the current SessionID, ResponseCode 'OK' and EVSENotification 'ReNegotiation' and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.4.2.2; 15118-2:8.4.4.2.3; 15118-2:8.8.4.1; 15118-2:8.8.4.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.8.4.2.2; 15118-2:8.6.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-839], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-513], [V2G2-848], [V2G2-242], [V2G2-516], [V2G2-680], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-689], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-840], [V2G2-841], [V2G2-849]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_EVCC_CMN_ReceiptRequired:= false, PICS_CMN_CMN_Renegotiation:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_005	

Table 702 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009'.

Table 702 — Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009'

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009
Test objective	Test System executes GoodCase procedure and initiates a renegotiation by sending a ChargingStatusRes message with the current SessionID, ResponseCode 'OK' and EVSENotification 'ReNegotiation' and all additional mandatory parameters, followed by a MeteringReceiptRes message. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.4.2.2; 15118-2:8.4.4.2.3; 15118-2:8.8.4.1; 15118-2:8.8.4.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.8.4.2.2; 15118-2:8.6.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-839], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-848], [V2G2-242], [V2G2-519], [V2G2-680], [V2G2-380], [V2G2-404], [V2G2-518], [V2G2-689], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-840], [V2G2-841], [V2G2-849]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_Renegotiation:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_006	

Table 703 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_010'.

Table 703 — Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_010'

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_010
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of PowerDeliveryRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	

Expected behavior
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_007

Table 704 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_011'.

Table 704 — Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_011'

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_011
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_008	

Table 705 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_012'.

Table 705 — Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_012'

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_012
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_008	

Table 706 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013'.

Table 706 — Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013'

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013
Test objective	Test System executes GoodCase procedure and waits for a renegotiation process initiated by the SUT during the charging loop. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.4.2.2; 15118-2:8.4.4.2.3; 15118-2:8.8.4.1; 15118-2:8.8.4.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.8.4.2.2; 15118-2:8.6.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-839], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-513], [V2G2-848], [V2G2-242], [V2G2-516], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-521], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-840], [V2G2-841], [V2G2-849]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= aC, PICS_EVCC_CMN_ReceiptRequired:= false PICS_CMN_CMN_Renegotiation:= false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_009	

Table 707 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014'.

Table 707 — Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014'

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014
Test objective	Test System executes GoodCase procedure and waits for a renegotiation process initiated by the SUT during the charging loop. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters after receiving a MeteringReceiptRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'ReNegotiation' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.4.2.2; 15118-2:8.4.4.2.3; 15118-2:8.8.4.1; 15118-2:8.8.4.2; 15118-2:8.4.3.9.1; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.8.4.2.2; 15118-2:8.6.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-839], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-848], [V2G2-242], [V2G2-519], [V2G2-380], [V2G2-404], [V2G2-518], [V2G2-522], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-840], [V2G2-841], [V2G2-849]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_Renegotiation:= false, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1

PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_010	

10.3.2.16 EVCC test cases for MeteringReceipt

The following subclauses include all *V2G MeteringReceipt message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.16.1 AC specific test cases

Table 708 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_001'.

Table 708 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_001'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_001
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.2.2; 15118-2:8.4.2; 15118-2:8.5.2.6; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-660]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_CMN_CMN_MeterReading:= unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 709 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_002'.

Table 709 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_002'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters.

	Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-517], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_002	

Table 710 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_003'.

Table 710 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_003'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-517], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_002	

Table 711 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_004'.

Table 711 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_004'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and

	terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-517], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_002	

Table 712 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_005'.

Table 712 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_005'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-517], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_002	

Table 713 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_006'.

Table 713 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_006'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within

	'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-517], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_002	

Table 714 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_007'.

Table 714 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_007'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_007
Test objective	Test System executes GoodCase procedure and sends a valid ChargingStatusRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-517], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_MeteringReceipt_001	

Table 715 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_008'.

Table 715 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_008'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_008
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo and all additional mandatory

	parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.4.2.2; 15118-2:8.4.2; 15118-2:8.5.2.6; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891], [V2G2-660]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certUpdate, PIXIT_CMN_CMN_MeterReading:= unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 716 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_009'.

Table 716 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_009'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_009
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.4.2.2; 15118-2:8.4.2; 15118-2:8.5.2.6; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896], [V2G2-660]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certInstall, PIXIT_CMN_CMN_MeterReading:= unknown,

PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= unknown
PreCondition
f_EVCC_AC_PR_ChargingStatus_001
Expected behavior
f_EVCC_AC_TB_VTB_MeteringReceipt_001

Table 717 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_010'.

Table 717 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_010'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_010
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of MeteringReceiptRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_MeteringReceipt_002	

Table 718 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_011'.

Table 718 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_011'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_011
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1

PreCondition
f_EVCC_AC_PR_MeteringReceipt_001
Expected behavior
f_EVCC_AC_TB_VTB_MeteringReceipt_003

Table 719 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_012'.

Table 719 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_012'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_012
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_003	

Table 720 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_013'.

Table 720 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_013'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_013
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.8.4.2.2; 15118-2:8.4.2; 15118-2:8.5.2.6; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-660], [V2G2-831]
Config Id	CF_04_002

PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_CMN_CMN_MeterReading:= meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 721 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_014'.

Table 721 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_014'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_014
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.4.2.2; 15118-2:8.4.2; 15118-2:8.5.2.6; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891], [V2G2-660], [V2G2-831]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certUpdate, PIXIT_CMN_CMN_MeterReading:= meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 722 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_015'.

Table 722 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_015'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_015
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message

	with a valid signature, current sessionID, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.4.2.2; 15118-2:8.4.2; 15118-2:8.5.2.6; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896], [V2G2-660], [V2G2-831]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certInstall, PIXIT_CMN_CMN_MeterReading:= meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 723 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_016'.

Table 723 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_016'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_016
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.4.2.2; 15118-2:8.4.2; 15118-2:8.5.2.6; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-660], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_CMN_CMN_MeterReading:= unknown,

	PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= sAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 724 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_017'.

Table 724 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_017'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_017
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.4.2.2; 15118-2:8.4.2; 15118-2:8.5.2.6; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891], [V2G2-660], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certUpdate, PIXIT_CMN_CMN_MeterReading:= unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= sAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 725 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_018'.

Table 725 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_018'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_018
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS

	Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.4.2.2; 15118-2:8.4.2; 15118-2:8.5.2.6; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896], [V2G2-660], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certInstall, PIXIT_CMN_CMN_MeterReading:= unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= sAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 726 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_019'.

Table 726 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_019'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_019
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.4.2.2; 15118-2:8.4.2; 15118-2:8.5.2.6; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-660], [V2G2-831], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_CMN_CMN_MeterReading:= meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= sAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	

f_EVCC_AC_TB_VTB_MeteringReceipt_001

Table 727 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_020'.

Table 727 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_020'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_020
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.4.2.2; 15118-2:8.4.2; 15118-2:8.5.2.6; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891], [V2G2-660], [V2G2-831], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certUpdate, PIXIT_CMN_CMN_MeterReading:= meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= sAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 728 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_021'.

Table 728 — Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_021'

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_021
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.4.2.2; 15118-2:8.4.2; 15118-2:8.5.2.6; 15118-2:8.6.2.3.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408],

	[V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896], [V2G2-660], [V2G2-831], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= aC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certInstall, PIXIT_CMN_CMN_MeterReading:= meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= sAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

10.3.2.16.2 DC specific test cases

Table 729 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_001'.

Table 729 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_001'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_001
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.8.4.2.3; 15118-2:8.5.2.6; 15118-2:8.4.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_CMN_CMN_MeterReading:= unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 730 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_002'.

Table 730 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_002'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-791], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_002	

Table 731 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_003'.

Table 731 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_003'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-791], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_002	

Table 732 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_004'.

Table 732 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_004'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'..
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-791], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_002	

Table 733 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_005'.

Table 733 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_005'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-791], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_002	

Table 734 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_006'.

Table 734 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_006'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_006

Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-791], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_002	

Table 735 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_007'.

Table 735 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_007'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_007
Test objective	Test System executes GoodCase procedure and sends a valid CurrentDemandRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-791], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_MeteringReceipt_001	

Table 736 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_008'.

Table 736 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_008'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_008

Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.2.3; 15118-2:8.5.2.6; 15118-2:8.4.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certUpdate, PIXIT_CMN_CMN_MeterReading:= unknown, PIXIT_EVCC_CMN_SAScheduleTypeIDMR:= unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 737 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_009'.

Table 737 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_009'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_009
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.2.3; 15118-2:8.5.2.6; 15118-2:8.4.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true,

	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certInstall, PIXIT_CMN_CMN_MeterReading:= unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 738 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_010'.

Table 738 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_010'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_010
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of MeteringReceiptRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_MeteringReceipt_002	

Table 739 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_011'.

Table 739 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_011'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_011
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC,

	PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_003	

Table 740 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_012'.

Table 740 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_012'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_012
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_003	

Table 741 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_013'.

Table 741 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_013'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_013
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.8.4.2.3; 15118-2:8.5.2.6; 15118-2:8.4.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766],

	[V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-831]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_CMN_CMN_MeterReading:= meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 742 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_014'.

Table 742 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_014'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_014
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.2.3; 15118-2:8.5.2.6; 15118-2:8.4.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891], [V2G2-831]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certUpdate, PIXIT_CMN_CMN_MeterReading:= meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 743 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_015'.

Table 743 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_015'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_015

Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.2.3; 15118-2:8.5.2.6; 15118-2:8.4.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896], [V2G2-831]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certInstall, PIXIT_CMN_CMN_MeterReading:= meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 744 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_016'.

Table 744 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_016'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_016
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.2.3; 15118-2:8.5.2.6; 15118-2:8.4.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC,

	PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_CMN_CMN_MeterReading:= unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= SAScheduleTupleID
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 745 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_017'.

Table 745 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_017'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_017
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.4.2.3; 15118-2:8.5.2.6; 15118-2:8.4.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certUpdate, PIXIT_CMN_CMN_MeterReading:= unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= SAScheduleTupleID
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 746 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_018'.

Table 746 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_018'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_018
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message

	with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.2.3; 15118-2:8.5.2.6; 15118-2:8.4.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certInstall, PIXIT_CMN_CMN_MeterReading:= unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= sAScheduleTupleID
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 747 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_019'.

Table 747 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_019'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_019
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.2.3; 15118-2:8.5.2.6; 15118-2:8.4.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-831], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_CMN_CMN_MeterReading:= meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= sAScheduleTupleID

PreCondition
f_EVCC_DC_PR_CurrentDemand_001
Expected behavior
f_EVCC_DC_TB_VTB_MeteringReceipt_001

Table 748 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_020'.

Table 748 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_020'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_020
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.2.3; 15118-2:8.5.2.6; 15118-2:8.4.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891], [V2G2-831], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certUpdate, PIXIT_CMN_CMN_MeterReading:= meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= sAScheduleTupleID
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 749 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_021'.

Table 749 — Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_021'

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_021
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.13.2; 15118-2:8.4.2.3; 15118-2:8.5.2.6; 15118-2:8.4.2; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-

	2:8.6.3.2; 15118-2:8.6.2.1; 15118-2:7.3.2; 15118-2:7.9.2.4.2; 15118-2:7.9.2.4.5; 15118-2:8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896], [V2G2-831], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1, PIXIT_EVCC_CMN_CertificateHandling:= certInstall, PIXIT_CMN_CMN_MeterReading:= meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR:= sAScheduleTupleID
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

10.3.2.17 EVCC test cases for CableCheck

The following subclauses include all *V2G CableCheck* message test case descriptions, where the EVCC is defined as SUT.

10.3.2.17.1 DC specific test cases

Table 750 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_001'.

Table 750 — Test case description for 'TC_EVCC_DC_VTB_CableCheck_001'

TC Id	TC_EVCC_DC_VTB_CableCheck_001
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT applies CP State C or D and sends a correct CableCheckReq message with the current SessionID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.5.2.2, 15118-2:8.5.4.2, 15118-2:8.7.3.1, 15118-2:8.7.3.6, 15118-2:8.7.4.4, 15118-2:8.8.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-249], [V2G2-250], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-912], [V2G2-599], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	

f_EVCC_DC_TB_VTB_CableCheck_001

Table 751 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_002'.

Table 751 — Test case description for 'TC_EVCC_DC_VTB_CableCheck_002'

TC Id	TC_EVCC_DC_VTB_CableCheck_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CableCheckRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-524], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_002	

Table 752 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_003'.

Table 752 — Test case description for 'TC_EVCC_DC_VTB_CableCheck_003'

TC Id	TC_EVCC_DC_VTB_CableCheck_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CableCheckRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-524], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_002	

Table 753 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_004'.

Table 753 — Test case description for 'TC_EVCC_DC_VTB_CableCheck_004'

TC Id	TC_EVCC_DC_VTB_CableCheck_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CableCheckRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-524], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_002	

Table 754 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_005'.

Table 754 — Test case description for 'TC_EVCC_DC_VTB_CableCheck_005'

TC Id	TC_EVCC_DC_VTB_CableCheck_005
Test objective	Test System executes GoodCase procedure, sends a valid ChargeParameterDiscoveryRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a CableCheckReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-524], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_003	

Table 755 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_006'.

Table 755 — Test case description for 'TC_EVCC_DC_VTB_CableCheck_006'

TC Id	TC_EVCC_DC_VTB_CableCheck_006
Test objective	Test System executes GoodCase procedure, sends continuously a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Ongoing',

	EVSEStatusCode 'EVSE_IsolationMonitoringActive' and all additional mandatory parameters and waits for another CableCheckReq message until the V2G_EVCC_CableCheck_Timer is equal or larger than V2G_EVCC_CableCheck_Timeout. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.3.1; 15118-2:8.7.3.6; 15118-2:8.8.4.2.3
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-605], [V2G2-700], [V2G2-701], [V2G2-702], [V2G2-617]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_004	

Table 756 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_007'.

Table 756 — Test case description for 'TC_EVCC_DC_VTB_CableCheck_007'

TC Id	TC_EVCC_DC_VTB_CableCheck_007
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of CableCheckRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_005	

Table 757 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_008'.

Table 757 — Test case description for 'TC_EVCC_DC_VTB_CableCheck_008'

TC Id	TC_EVCC_DC_VTB_CableCheck_008
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as

	soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_006	

Table 758 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_009'

Table 758 — Test case description for 'TC_EVCC_DC_VTB_CableCheck_009'

TC Id	TC_EVCC_DC_VTB_CableCheck_009
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_006	

10.3.2.18 EVCC test cases for PreCharge

The following subclause includes all *V2G PreCharge message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.18.1 DC specific test cases

Table 759 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_001'.

Table 759 — Test case description for 'TC_EVCC_DC_VTB_PreCharge_001'

TC Id	TC_EVCC_DC_VTB_PreCharge_001
Test objective	Test System executes GoodCase procedure and sends a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters.

	Test System then checks that the SUT sends a correct PreChargeReq message with the current SessionID, target voltage and current and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.5.4; 15118-2:8.7.3.1; 15118-2:8.7.3.7; 15118-2:8.5.2.7; 15118-2:8.4.5.3; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-253], [V2G2-254], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-525], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_001	

Table 760 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_002'.

Table 760 — Test case description for 'TC_EVCC_DC_VTB_PreCharge_002'

TC Id	TC_EVCC_DC_VTB_PreCharge_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PreChargeRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-526], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_002	

Table 761 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_003'.

Table 761 — Test case description for 'TC_EVCC_DC_VTB_PreCharge_003'

TC Id	TC_EVCC_DC_VTB_PreCharge_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PreChargeRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within

	'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-526], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_002	

Table 762 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_004'.

Table 762 — Test case description for 'TC_EVCC_DC_VTB_PreCharge_004'

TC Id	TC_EVCC_DC_VTB_PreCharge_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PreChargeRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-526], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_002	

Table 763 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_005'.

Table 763 — Test case description for 'TC_EVCC_DC_VTB_PreCharge_005'

TC Id	TC_EVCC_DC_VTB_PreCharge_005
Test objective	Test System executes GoodCase procedure, sends a valid CableCheckRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a PreChargeReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2

Referenced requirement(s)	[V2G2-526], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_003	

Table 764 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_006'.

Table 764 — Test case description for 'TC_EVCC_DC_VTB_PreCharge_006'

TC Id	TC_EVCC_DC_VTB_PreCharge_006
Test objective	Test System executes GoodCase procedure and sends continuously a PreChargeRes message with the current SessionID, ResponseCode 'OK', EVSEPresentVoltage '0' and all additional mandatory parameters and waits for another PreChargeReq message with the current SessionID, target voltage and current and all additional mandatory parameters until the V2G_EVCC_PreCharge_Timer is equal or larger than V2G_EVCC_PreCharge_Timeout. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.3.1; 15118-2:8.7.3.7; 15118-2:8.8.4.2.3
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-605], [V2G2-704], [V2G2-705], [V2G2-706], [V2G2-832], [V2G2-618]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_004	

Table 765 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_007'.

Table 765 — Test case description for 'TC_EVCC_DC_VTB_PreCharge_007'

TC Id	TC_EVCC_DC_VTB_PreCharge_007
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of PreChargeRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]

Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_005	

Table 766 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_008'.

Table 766 — Test case description for 'TC_EVCC_DC_VTB_PreCharge_008'

TC Id	TC_EVCC_DC_VTB_PreCharge_008
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' present voltage and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_006	

Table 767 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_009'.

Table 767 — Test case description for 'TC_EVCC_DC_VTB_PreCharge_009'

TC Id	TC_EVCC_DC_VTB_PreCharge_009
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	

PreCondition
f_EVCC_DC_PR_PreCharge_001
Expected behavior
f_EVCC_DC_TB_VTB_PreCharge_006

10.3.2.19 EVCC test cases for PreChargeOrPowerDelivery

The following subclause includes all V2G PreChargeOrPowerDelivery *message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.19.1 DC specific test cases

Table 768 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001'.

Table 768 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001
Test objective	Test System executes GoodCase procedure and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.4.4.2.3; 15118-2:8.4.5.3; 15118-2:8.4.3.9.1; 15118-2:8.5.4.2; 15118-2:8.5.4.5; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-293], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_CMN_ChargingProfileOptimization:= unknown
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_001	

Table 769 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_002'.

Table 769 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_002'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within

	'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002	

Table 770 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_003'.

Table 770 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_003'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002	

Table 771 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_004'.

Table 771 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_004'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;

Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002	

Table 772 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_005'.

Table 772 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_005'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ChargingProfileInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002	

Table 773 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_006'.

Table 773 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_006'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_TariffSelectionInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002

PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002	

Table 774 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_007'.

Table 774 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_007'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_007
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_PowerDeliveryNotApplied' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002	

Table 775 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_008'.

Table 775 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_008'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_008
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ContactorError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	

f_EVCC_DC_PR_PreChargeOrPowerDelivery_001
Expected behavior
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002

Table 776 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009'.

Table 776 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009
Test objective	Test System executes GoodCase procedure, sends a valid PreChargeRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_003	

Table 777 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_010'.

Table 777 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_010'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_010
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of PowerDeliveryRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_004	

Table 778 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_011'.

Table 778 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_011'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_011
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_005	

Table 779 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_012'.

Table 779 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_012'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_012
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_005	

Table 780 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013'.

Table 780 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013

Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.4.5.3; 15118-2:8.4.3.9.1; 15118-2:8.5.4.2; 15118-2:8.5.4.5; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_Renegotiation:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization:= unknown, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

Table 781 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014'.

Table 781 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.4.5.3; 15118-2:8.4.3.9.1; 15118-2:8.5.4.2; 15118-2:8.5.4.5; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]

Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_Renegotiation:= false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_CMN_ChargingProfileOptimization:= unknown, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

Table 782 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015'.

Table 782 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015
Test objective	Test System executes GoodCase procedure and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired (A valid ChargeParameterDiscoveryRes message with a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs was sent before). Test System then checks that the SUP sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.4.5.3; 15118-2:8.4.3.9.1; 15118-2:8.5.4.2; 15118-2:8.5.4.5; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-293], [V2G2-296], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_CMN_ChargingProfileOptimization:= optimization
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_001	

Table 783 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016'.

Table 783 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016
Test objective	Test System executes GoodCase procedure and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If

	another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired (A valid ChargeParameterDiscoveryRes message with a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs was sent before). Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including the default SAScheduleTuple and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.4.5.3; 15118-2:8.4.3.9.1; 15118-2:8.5.4.2; 15118-2:8.5.4.5; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_CMN_ChargingProfileOptimization:= none_
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_001	

Table 784 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017'.

Table 784 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired (A valid ChargeParameterDiscoveryRes message with a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs was sent within renegotiation before). Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.4.5.3; 15118-2:8.4.3.9.1; 15118-2:8.5.4.2; 15118-2:8.5.4.5; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659],

	[V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-296], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_Renegotiation:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization:= optimization, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

Table 785 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018'.

Table 785 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge.Timeout' has expired (A valid ChargeParameterDiscoveryRes message with a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs was sent within renegotiation before). Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.4.5.3; 15118-2:8.4.3.9.1; 15118-2:8.5.4.2; 15118-2:8.5.4.5; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-296], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_Renegotiation:= false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_CMN_ChargingProfileOptimization:= optimization, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

Table 786 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019'.

Table 786 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired (A valid ChargeParameterDiscoveryRes message with a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs was sent within renegotiation before). Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including the default SAScheduleTuple and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.4.4.2.3; 15118-2:8.4.5.3; 15118-2:8.4.3.9.1; 15118-2:8.5.4.2; 15118-2:8.5.4.5; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_Renegotiation:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization:= none_, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

Table 787 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020'.

Table 787 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired (A valid ChargeParameterDiscoveryRes message with a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs was sent within renegotiation before). Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including the default

	SAScheduleTuple and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.4.5.3; 15118-2:8.4.3.9.1; 15118-2:8.5.4.2; 15118-2:8.5.4.5; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.5.2.12; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_Renegotiation:= false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_CMN_ChargingProfileOptimization:= none_, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= unknown
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

Table 788 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021'.

Table 788 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', the previous charging profile including the same SAScheduleTupleID than before renegotiation and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.8.4.2.3; 15118-2:8.4.5.3; 15118-2:8.4.3.9.1; 15118-2:8.5.4.2; 15118-2:8.5.4.5; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_Renegotiation:= true

PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization:= unknown, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= newSAScheduleTupleID
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

Table 789 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022'.

Table 789 — Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022'

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', the previous charging profile including the same SAScheduleTupleID than before renegotiation and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.4.2.3; 15118-2:8.4.5.3; 15118-2:8.4.3.9.1; 15118-2:8.5.4.2; 15118-2:8.5.4.5; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.5.2.10; 15118-2:8.5.2.11; 15118-2:8.7.4.2; 15118-2:8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_Renegotiation:= false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_EVCC_CMN_ChargingProfileOptimization:= unknown, PIXIT_EVCC_CMN_NewSAScheduleTupleID:= newSAScheduleTupleID
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

10.3.2.20 EVCC test cases for CurrentDemand

The following subclause includes all *V2G CurrentDemand message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.20.1 DC specific test cases

Table 790 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_001'.

Table 790 — Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_001'

TC Id	TC_EVCC_DC_VTB_CurrentDemand_001
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', and all additional mandatory parameters. Test System then checks that the SUT sends a correct CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.5.4.2; 15118-2:8.4.5.4.3; 15118-2:8.5.4.2; 15118-2:8.4.2; 15118-2:8.4.4.2; 15118-2:8.5.2.7; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-257], [V2G2-258], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-530], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_001	

Table 791 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_002'.

Table 791 — Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_002'

TC Id	TC_EVCC_DC_VTB_CurrentDemand_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CurrentDemandRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.8.4.2.3
Referenced requirement(s)	[V2G2-532], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_002	

Table 792 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_003'.

Table 792 — Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_003'

TC Id	TC_EVCC_DC_VTB_CurrentDemand_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CurrentDemandRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.8.4.2.3
Referenced requirement(s)	[V2G2-532], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_002	

Table 793 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_004'.

Table 793 — Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_004'

TC Id	TC_EVCC_DC_VTB_CurrentDemand_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CurrentDemandRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.8.4.2.3
Referenced requirement(s)	[V2G2-532], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_002	

Table 794 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_005'.

Table 794 — Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_005'

TC Id	TC_EVCC_DC_VTB_CurrentDemand_005
Test objective	Test System executes GoodCase procedure, sends valid PowerDeliveryRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a CurrentDemandReq message with the current SessionID, target voltage and current

	and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.8.4.2.3; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-532], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_003	

Table 795 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_006'.

Table 795 — Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_006'

TC Id	TC_EVCC_DC_VTB_CurrentDemand_006
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_004	

Table 796 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_007'.

Table 796 — Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_007'

TC Id	TC_EVCC_DC_VTB_CurrentDemand_007
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as

	soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_004	

Table 797 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_008'.

Table 797 — Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_008'

TC Id	TC_EVCC_DC_VTB_CurrentDemand_008
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of CurrentDemandRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_005	

10.3.2.21 EVCC test cases for ChargingLoopDcOrPowerDelivery

The following subclause includes all *V2G ChargingLoopDcOrPowerDelivery message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.21.1 DC specific test cases

Table 798 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001'.

Table 798 — Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001'

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001
Test objective	Test System executes GoodCase procedure and sends a number (PICS_CMN_LoopCounter) of CurrentDemandRes messages with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. In the last sequence EVSENotification will be changed to 'StopCharging'.

	Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.4.5.4.3; 15118-2:8.5.4.2; 15118-2:8.4.2; 15118-2:8.4.3.9.1; 15118-2:8.8.4.1; 15118-2:8.7.4.2; 15118-2:8.5.4.5; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.8.4.2.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-839], [V2G2-278], [V2G2-279], [V2G2-257], [V2G2-258], [V2G2-286], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-527], [V2G2-531], [V2G2-679], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-293], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_EVCC_CMN_ReceiptRequired:= false
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_001	

Table 799 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002'.

Table 799 — Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002'

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002
Test objective	Test System executes GoodCase procedure and sends a number (PICS_CMN_LoopCounter) of CurrentDemandRes messages with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters, always followed by MeteringReceiptRes messages. In the last charging loop sequence EVSENotification will be changed to 'StopCharging'. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.4.5.4.3; 15118-2:8.5.4.2; 15118-2:8.4.2; 15118-2:8.4.3.9.1; 15118-2:8.8.4.1; ; 15118-2:8.7.4.2; 15118-2:8.5.4.5; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.8.4.2.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-839], [V2G2-278], [V2G2-279], [V2G2-257], [V2G2-258], [V2G2-286], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-793], [V2G2-679], [V2G2-394], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-792], [V2G2-832], [V2G2-903], [V2G2-293], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	

f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_002

Table 800 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_003'.

Table 800 — Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_003'

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003	

Table 801 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_004'.

Table 801 — Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_004'

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003	

Table 802 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_005'.

Table 802 — Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_005'

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003	

Table 803 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_006'.

Table 803 — Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_006'

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ContactorError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003	

Table 804 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007'.

Table 804 — Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007'

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007
Test objective	Test System executes GoodCase procedure, sends a valid CurrentDemandRes or MeteringReceiptRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a PowerDeliveryReq message with the current SessionID,

	ChargeProgress 'Stop_' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_004	

Table 805 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008'.

Table 805 — Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008'

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008
Test objective	Test System executes GoodCase procedure and initiates a renegotiation by sending a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'Renegotiation' and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.4.5.4.3; 15118-2:8.5.4.2; 15118-2:8.4.2; 15118-2:8.4.3.9.1; 15118-2:8.8.4.1; 15118-2:8.5.4.5; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-257], [V2G2-258], [V2G2-286], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-531], [V2G2-680], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-840], [V2G2-841], [V2G2-832], [V2G2-686]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_EVCC_CMN_ReceiptRequired:= false, PICS_CMN_CMN_Renegotiation:= true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_005	

Table 806 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009'.

Table 806 — Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009'

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009

Test objective	Test System executes GoodCase procedure and initiates a renegotiation by sending a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'Renegotiation' and all additional mandatory parameters, followed by a MeteringReceiptRes message. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.4.5.4.3; 15118-2:8.5.4.2; 15118-2:8.4.2; 15118-2:8.4.3.9.1; 15118-2:8.8.4.1; 15118-2:8.5.4.5; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.8.4.2.3; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-257], [V2G2-258], [V2G2-286], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-680], [V2G2-394], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-792], [V2G2-840], [V2G2-841], [V2G2-832], [V2G2-794]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_Renegotiation:= true, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= unknown, PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_006	

Table 807 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_010'.

Table 807 — Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_010'

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_010
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of PowerDeliveryRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_007	

Table 808 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_011'.

Table 808 — Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_011'

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_011
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_008	

Table 809 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_012'.

Table 809 — Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_012'

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_012
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_008	

Table 810 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013'.

Table 810 — Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013'

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013
Test objective	Test System executes GoodCase procedure and waits for a renegotiation process initiated by the SUT during the charging loop.

	Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.4.5.4.3; 15118-2:8.5.4.2; 15118-2:8.4.2; 15118-2:8.4.3.9.1; 15118-2:8.8.4.1; 15118-2:8.5.4.5; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-257], [V2G2-258], [V2G2-286], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-531], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-840], [V2G2-841], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_EVCC_CMN_ReceiptRequired:= false, PICS_CMN_CMN_Renegotiation:= false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_009	

Table 811 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014'.

Table 811 — Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014'

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014
Test objective	Test System executes GoodCase procedure and waits for a renegotiation process initiated by the SUT during the charging loop. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.9.2, 15118-2:8.4.3.9.3; 15118-2:8.4.5.4.3; 15118-2:8.5.4.2; 15118-2:8.4.2; 15118-2:8.4.3.9.1; 15118-2:8.8.4.1; 15118-2:8.5.4.5; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.8.4.2.3; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-257], [V2G2-258], [V2G2-286], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-394], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-792], [V2G2-840], [V2G2-841], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode:= pnC, PICS_EVCC_CMN_ReceiptRequired:= true, PICS_CMN_CMN_Renegotiation:= false, PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_PkiName:= PKI_1
PIXIT selection	PIXIT_CMN_CMN_Renegotiation:= renegotiation, PIXIT_CMN_CMN_PkiName:= PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_010	

10.3.2.22 EVCC test cases for WeldingDetection

The following subclause includes all *V2G WeldingDetection message* test case descriptions, where the EVCC is defined as SUT.

10.3.2.2.1 DC specific test cases

Table 812 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_001'.

Table 812 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_001'

TC Id	TC_EVCC_DC_VTB_WeldingDetection_001
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B and sends a correct WeldingDetectionReq message with the current SessionID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.5.5.2; 15118-2:8.4.2.3; 15118-2:8.4.2; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-261], [V2G2-262], [V2G2-533], [V2G2-747], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetetction
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_001	

Table 813 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_002'.

Table 813 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_002'

TC Id	TC_EVCC_DC_VTB_WeldingDetection_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a WeldingDetectionRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-534], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetetction
PreCondition	
f_EVCC_DC_PR_WeldingDetection_001	
Expected behavior	

f_EVCC_DC_TB_VTB_WeldingDetection_002

Table 814 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_003'.

Table 814 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_003'

TC Id	TC_EVCC_DC_VTB_WeldingDetection_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a WeldingDetectionRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-534], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetetction
PreCondition	
f_EVCC_DC_PR_WeldingDetection_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_002	

Table 815 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_004'.

Table 815 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_004'

TC Id	TC_EVCC_DC_VTB_WeldingDetection_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a WeldingDetectionRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1;
Referenced requirement(s)	[V2G2-534], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetetction
PreCondition	
f_EVCC_DC_PR_WeldingDetection_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_002	

Table 816 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_005'.

Table 816 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_005'

TC Id	TC_EVCC_DC_VTB_WeldingDetection_005
Test objective	Test System executes GoodCase procedure, sends a valid PowerDeliveryRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a WeldingDetectionReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-534], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetetction
PreCondition	
f_EVCC_DC_PR_WeldingDetection_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_003	

Table 817 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_006'.

Table 817 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_006'

TC Id	TC_EVCC_DC_VTB_WeldingDetection_006
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of WeldingDetectionRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetetction
PreCondition	
f_EVCC_DC_PR_WeldingDetection_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_004	

Table 818 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_007'.

Table 818 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_007'

TC Id	TC_EVCC_DC_VTB_WeldingDetection_007
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.

	Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetetction
PreCondition	
f_EVCC_DC_PR_WeldingDetection_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_005	

Table 819 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_008'.

Table 819 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_008'

TC Id	TC_EVCC_DC_VTB_WeldingDetection_008
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetection
PreCondition	
f_EVCC_DC_PR_WeldingDetection_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_005	

Table 820 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_009'.

Table 820 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_009'

TC Id	TC_EVCC_DC_VTB_WeldingDetection_009
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B and sends a correct WeldingDetectionReq message with the current SessionID and all additional mandatory parameters. Furthermore the repetition of the WeldingDetection message sequence will be checked.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.5.5.2; 15118-2:8.4.2.3; 15118-2:8.4.2; 15118-2:8.5.4.2; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.8.4.2.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-261], [V2G2-262], [V2G2-533], [V2G2-747], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-620]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= weldingDetetction
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_006	

10.3.2.23 EVCC test cases for WeldingDetectionOrSessionStop

The following subclause includes all *V2G WeldingDetectionOrSessionStop* message test case descriptions, where the EVCC is defined as SUT.

10.3.2.23.1 DC specific test cases

Table 821 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001'.

Table 821 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.12; 15118-2:8.4.5.5; 15118-2:8.4.1; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= unknown, PIXIT_EVCC_CMN_TLS:= tls, PIXIT_EVCC_CMN_Pause:= unknown

PreCondition
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001
Expected behavior
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001

Table 822 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_002'.

Table 822 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_002'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionStopRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_004	

Table 823 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_003'.

Table 823 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_003'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionStopRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001	
Expected behavior	

f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_004

Table 824 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_004'.

Table 824 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_004'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionStopRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_004	

Table 825 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005'.

Table 825 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005
Test objective	Test System executes GoodCase procedure, sends a valid sends a PowerDeliveryRes or WeldingDetectionRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving the SessionStopReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.8.4.2.1; 15118-2:8.7.2.1; 15118-2:8.7.2.2
Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_005	

Table 826 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_006'.

Table 826 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_006'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_006
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of SessionStopRes message. Test System then checks that the SUT applies CP State B (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:7.4; 15118-2:8.2.1; 15118-2:8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_006	

Table 827 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007'.

Table 827 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.12.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.3.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-633], [V2G2-635], [V2G2-637]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= unknown, PIXIT_EVCC_CMN_TLS:= none_, PIXIT_EVCC_CMN_Pause:= unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	

Expected behavior
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001

Table 828 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008'.

Table 828 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.12.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= unknown, PIXIT_EVCC_CMN_TLS:= unknown, PIXIT_EVCC_CMN_Pause:= unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 829 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009'.

Table 829 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and

	Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.12; 15118-2:8.4.5.5; 15118-2:8.4.1; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= unknown, PIXIT_EVCC_CMN_TLS:= tls, PIXIT_EVCC_CMN_Pause:= none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 830 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010'.

Table 830 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.12.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.3.4; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= unknown, PIXIT_EVCC_CMN_TLS:= none_, PIXIT_EVCC_CMN_Pause:= none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 831 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011'.

Table 831 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.12.1; 15118-2:8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= unknown, PIXIT_EVCC_CMN_TLS:= unknown, PIXIT_EVCC_CMN_Pause:= none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 832 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012'.

Table 832 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.12; 15118-2:8.4.5.5; 15118-2:8.4.1; 15118-2:8.8.1; 15118-

	2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= unknown, PIXIT_EVCC_CMN_TLS:= tls, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 833 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013'.

Table 833 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.12.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.3.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= unknown, PIXIT_EVCC_CMN_TLS:= none_, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 834 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014'.

Table 834 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.12.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= unknown, PIXIT_EVCC_CMN_TLS:= unknown, PIXIT_EVCC_CMN_Pause:= pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 835 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015'.

Table 835 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.

Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.12; 15118-2:8.4.5.5; 15118-2:8.4.1; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= unknown, PIXIT_EVCC_CMN_TLS:= tls
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 836 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016'.

Table 836 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.12.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.3.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= unknown, PIXIT_EVCC_CMN_TLS:= none_

PreCondition
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001
Expected behavior
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001

Table 837 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017'.

Table 837 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.12.1; 15118-2:8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= unknown, PIXIT_EVCC_CMN_TLS:= unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 838 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018'.

Table 838 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct

	SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.4.3.12; 15118-2:8.4.5.5; 15118-2:8.4.1; 15118-2:8.8.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.2.4.2; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= unknown, PIXIT_EVCC_CMN_TLS:= tls
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 839 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019'.

Table 839 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.12.1; 15118-2:8.8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.3.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-

	718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= unknown, PIXIT_EVCC_CMN_TLS:= none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 840 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020'.

Table 840 — Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020'

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_TP_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2:8.3.2; 15118-2:8.3.3; 15118-2:8.3.4; 15118-2:8.4.2; 15118-2:8.4.3.12.1; 15118-2:8.4.2.3; 15118-2:8.6.2.4.1; 15118-2:8.6.3.1; 15118-2:8.6.2.1; 15118-2:8.7.4.4; 15118-2:7.4; 15118-2:8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode:= dC, PICS_CMN_CMN_IdentificationMode:= eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow:= sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection:= unknown, PIXIT_EVCC_CMN_TLS:= unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Annex A (normative)

Configuration specifications

This annex covers all test configuration related information in the ATS.

A.1 Timer configuration

```

module Timer_15118 {
    group Timer_par_15118 {
        // ----- Non-standardised CMN Processing timeouts -----
        modulepar float par_CMN_Transmission_Delay := 0.25;
        modulepar float par_CMN_HAL_Timeout := 2.0;
        modulepar float par_CMN_waitForNextHAL := 0.5;
        modulepar float par_CMN_Intern_PTC_Timeout := 0.4;
        modulepar float par_CMN_setKey := 3.0;

        // ----- Non-standardised EVCC Processing timeouts -----
        modulepar float par_EVCC_waitForNextTC := 2.0;
        // ----- Non-standardised SECC Processing timeouts -----

        modulepar float par_SECC_waitForNextTC := 20.0;
        modulepar float par_SECC_T_step_X1 := 3.0;

        modulepar float par_SECC_Pmax0W := 200.0;
    }
}

module Timer_15118_2 {
    group Timer_par_15118_2 {
        //----- SECC message performance timeouts -----
        modulepar float par_V2G_SECC_Sequence_Timeout := 60.0;

        //----- EVCC message timeouts -----
        modulepar float par_V2G_EVCC_Msg_Timeout_SupportedAppProtocolReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_SessionSetupReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_ServiceDetailReq := 5.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_CertificateInstallationReq := 5.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq := 5.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_PaymentDetailsReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_AuthorizationReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq := 5.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_ChargingStatusReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_MeteringReceiptReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_SessionStopReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_CableCheckReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_PreChargeReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_CurrentDemandReq := 0.25;
        modulepar float par_V2G_EVCC_Msg_Timeout_WeldingDetectionReq := 2.0;

        //----- EVCC Processing timeouts -----
        modulepar float par_V2G_EVCC_Ongoing_Timeout := 60.0;
        modulepar float par_V2G_EVCC_Ongoing_Performance_Timeout := 55.0;

        //----- Sequence CommunicationSetup timeouts -----
        modulepar float par_V2G_EVCC_CommunicationSetup_Timeout := 20.0;
        modulepar float par_V2G_SECC_CommunicationSetup_Performance_Time := 18.0;

        // ----- Sequence CableCheck timeouts -----
        modulepar float par_V2G_EVCC_CableCheck_Timeout := 40.0;
    }
}

```



```

// ----- Sequence PreCharge timeouts -----
modulepar float par_V2G_EVCC_PreCharge_Timeout := 7.0;

// ----- SDP timeout -----
modulepar float par_EVCC_SDP_Timeout := 0.25;

// ----- Non-standardised SECC Processing timeouts -----
modulepar float par_SECC_B1_B2_Timeout := 8.0;
modulepar float par_SECC_chargingLoop_pause := 0.1;

// ----- Non-standardised EVCC Processing timeouts -----
modulepar float par_V2G_SECC_CPState_Error_Detection_Timeout := 1.5;
modulepar float par_V2G_SECC_CPState_Detection_Timeout := 0.25;

// ----- Non-standardised CMN Processing timeouts -----
modulepar float par_CMN_TCP_Connection_Termination_Timeout := 5.0;
}
}

```

A.2 PICS configuration

```

module Pics_15118 {

    import from DataStructure_PICS_15118 all;

    group PICS_15118 {

        group PICS_CMN {

            modulepar boolean PICS_CMN_CMN_CombinedTesting := false;

            modulepar ChargingMode PICS_CMN_CMN_ChargingMode := aC;

            modulepar IdentificationMode PICS_CMN_CMN_IdentificationMode := eIM;

            modulepar PlugType PICS_CMN_CMN_PlugType := type2;

            modulepar CableCapabilityACType PICS_CMN_AC_CableCapability := capability32A;

            modulepar float PICS_CMN_CMN_WakeUp := 300.0;
        }

        group PICS_SECC_Tester {

            modulepar boolean PICS_SECC_CMN_Pause := false;

            modulepar EIMDone PICS_SECC_CMN_EIMDone := v2gAuthorization;
        }

        group PICS_EVCC_Tester {

            modulepar ZeroPow PICS_EVCC_CMN_PmaxSchedulewithZeroPow := none_;
        }
    }
}

```

```

module Pics_15118_2 {

    import from DataStructure_HAL_61851 all;
    import from DataStructure_PICS_15118_2 all;
    import from DataStructure_Security all;

    group PICS_15118_2 {

        group PICS_CMN {

            modulepar boolean PICS_CMN_CMN_V2gtpSdp := true;

            modulepar boolean PICS_CMN_CMN_V2gtpV2g := true;
        }
    }
}

```

```

modulepar boolean PICS_CMN_CMN_Renegotiation := false;

modulepar integer PICS_CMN_CMN_RenegotiationLoopIndication := 10;

modulepar integer PICS_CMN_CMN_LoopCounter := 8;

modulepar boolean PICS_CMN_CMN_InvalidV2gtpDataFields := true;

modulepar boolean PICS_CMN_CMN_SDPTimeoutsAndInvalidData := true;

modulepar boolean PICS_CMN_CMN_FailedResponseCodes := true;

modulepar boolean PICS_CMN_CMN_V2gTimeouts := true;

modulepar PkiType PICS_CMN_CMN_PkiName := PKI_1;
}

group PICS_SECC_Tester {

    modulepar IEC_61851_States PICS_SECC_CMN_EvStateInChargeProgress := C;

    modulepar boolean PICS_SECC_DC_WeldingDetection := true;

    modulepar boolean PICS_SECC_CMN_DepartureTime := false;

    modulepar boolean PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false;
}

group PICS_EVCC_Tester {

    modulepar boolean PICS_EVCC_CMN_ReceiptRequired := true;

    modulepar boolean PICS_EVCC_CMN_InvalidStatesAndDutyCycles := true;

    modulepar boolean PICS_EVCC_CMN_InvalidV2GMessages := true;

    modulepar boolean PICS_CMN_EVCC_SalesTariff := true;
}
}
}

```

A.3 PIXIT configuration

```

module Pixit_15118 {

    import from DataStructure_PIXIT_15118 all;

    group PIXIT_15118 {

        group PIXIT_CMN {

            modulepar float PIXIT_CMN_CMN_WakeUp := 200.0;
        }

        group PIXIT_EVCC_Tester {

            modulepar Pause PIXIT_EVCC_CMN_Pause := unknown;
        }
    }
}

```

```

module Pixit_15118_2 {

    import from DataStructure_PIXIT_15118_2 all;

    group PIXIT_15118_2 {

        group PIXIT_CMN {

```

```

    modulepar PkiType PIXIT_CMN_CMN_PkiName := PKI_1;

    modulepar Renegotiation PIXIT_CMN_CMN_Renegotiation := unknown;

    modulepar MeterReading PIXIT_CMN_CMN_MeterReading := unknown;
}

group PIXIT_SECC_Tester {

    modulepar VASSECC PIXIT_SECC_CMN_VAS := none_;

    modulepar boolean PIXIT_SECC_CMN_ReceiptRequired := true;

    modulepar CertificateHandlingSECC PIXIT_SECC_CMN_CertificateHandling := none_;

    modulepar SalesTariff PIXIT_SECC_CMN_SalesTariff := unknown;

    modulepar boolean PIXIT_SECC_CMN_TLS := false;

    modulepar integer PIXIT_SECC_CMN_VersionNumberMinor := 0;

    modulepar InternetAccess PIXIT_SECC_CMN_InternetAccess := unknown;

    modulepar Ongoing PIXIT_SECC_DC_CableCheckOngoing := unknown;

    modulepar Ongoing PIXIT_SECC_CMN_AuthorizationOngoing := unknown;

    modulepar Ongoing PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing := unknown;

    modulepar MeterInfo PIXIT_SECC_CMN_MeterInfo := unknown;

    modulepar PaymentOption PIXIT_SECC_CMN_PaymentOption := externalPayment;
}

group PIXIT_EVCC_Tester {

    modulepar VASEVCC PIXIT_EVCC_CMN_VAS := unknown;

    modulepar WeldingDetection PIXIT_EVCC_DC_WeldingDetection := unknown;

    modulepar TLS PIXIT_EVCC_CMN_TLS := unknown;

    modulepar CertificateHandlingEVCC PIXIT_EVCC_CMN_CertificateHandling := unknown;

    modulepar DepartureTime PIXIT_EVCC_CMN_DepartureTime := unknown;

    modulepar MaxNumberOfPhasesInUse PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown;

    modulepar ChargingProfileOpt PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown;

    modulepar NewSAScheduleTupleID PIXIT_EVCC_CMN_NewSAScheduleTupleID := unknown;

    modulepar SAScheduleTupleIDMR PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown;
}
}

```

Annex B (normative)

Control part specification

This annex covers all control part specifications in the ATS.

B.1 SECC control parts

B.1.1 AC specific control parts

```

module AC_SECC_Error_Control {

    import from Pics_15118_2 all;
    import from Pixit_15118_2 all;
    import from Pics_15118 all;
    import from TestCases_SECC_V2GTPSDP all;
    import from TestCases_SECC_SDP all;
    import from TestCases_SECC_V2GTPSessionSetup all;
    import from TestCases_SECC_SessionSetup all;
    import from TestCases_SECC_SupportedAppProtocol all;
    import from TestCases_SECC_ServiceDiscovery all;
    import from TestCases_SECC_ServiceDetail all;
    import from TestCases_SECC_PaymentServiceSelection all;
    import from TestCases_SECC_CertificateInstallation all;
    import from TestCases_SECC_CertificateUpdate all;
    import from TestCases_SECC_PaymentDetails all;
    import from TestCases_SECC_Authorization all;
    import from TestCases_SECC_ServiceDetail all;
    import from TestCases_SECC_ChargeParameterDiscovery all;
    import from TestCases_SECC_PowerDelivery all;
    import from TestCases_SECC_ChargingStatus all;
    import from TestCases_SECC_MeteringReceipt all;
    import from TestCases_SECC_SessionStop all;
    import from ComponentsAndPorts all;

    control {

        /** Test Group 1: Invalid V2GTP data fields */

        if (PICS_CMN_CMN_InvalidV2gtpDataFields) {

            // V2GTP explicit and SDP ++++++
            if (PICS_CMN_CMN_V2gtpSdp) { execute(TC_SECC_CMN_VTB_V2GTPSDP_002()); }

            if (PICS_CMN_CMN_V2gtpSdp) { execute(TC_SECC_CMN_VTB_V2GTPSDP_003()); }

            if (PICS_CMN_CMN_V2gtpSdp) { execute(TC_SECC_CMN_VTB_V2GTPSDP_004()); }

            if (PICS_CMN_CMN_V2gtpSdp) { execute(TC_SECC_CMN_VTB_V2GTPSDP_005()); }

            // V2GTP explicit and V2G (SessionSetup) ++++++
            if (PICS_CMN_CMN_V2gtpV2g) { execute(TC_SECC_CMN_VTB_V2GTPSessionSetup_002()); }

            if (PICS_CMN_CMN_V2gtpV2g) { execute(TC_SECC_CMN_VTB_V2GTPSessionSetup_003()); }

            if (PICS_CMN_CMN_V2gtpV2g) { execute(TC_SECC_CMN_VTB_V2GTPSessionSetup_004()); }

            if (PICS_CMN_CMN_V2gtpV2g) { execute(TC_SECC_CMN_VTB_V2GTPSessionSetup_005()); }

        }

        /** Test Group 2: SDP timeouts and invalid SDP data fields */

        if (PICS_CMN_CMN_SDPTimeoutsAndInvalidData) {

            // SDP ++++++
            if (PICS_CMN_CMN_IdentificationMode == eIM and not
                PIXIT_SECC_CMN_TLS) { execute(TC_SECC_CMN_VTB_SDP_004()); }

        }

    }

}

```

```

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_SECC_CMN_TLS) {execute(TC_SECC_CMN_VTB_SDP_005());};

    if (PICS_CMN_CMN_IdentificationMode == pnC) {execute(TC_SECC_CMN_VTB_SDP_006());};
}

/** Test Group 3: V2G FAILED ResponseCodes **/

if (PICS_CMN_CMN_FailedResponseCodes) {

    // SupportedAppProtocol ++++++//
    if (true){ execute(TC_SECC_CMN_VTB_SupportedAppProtocol_002());};

    // SessionSetup ++++++//
    if (true){ execute(TC_SECC_CMN_VTB_SessionSetup_002());};

    // ServiceDiscovery ++++++//
    if (true){ execute(TC_SECC_CMN_VTB_ServiceDiscovery_002());};

    if (true){ execute(TC_SECC_CMN_VTB_ServiceDiscovery_003());};

    // ServiceDetail ++++++//
    if (PIXIT_SECC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_SECC_CMN_TLS == true){ execute(TC_SECC_CMN_VTB_ServiceDetail_003());};

    if (PIXIT_SECC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_SECC_CMN_TLS == true){ execute(TC_SECC_CMN_VTB_ServiceDetail_005());};

    if (PIXIT_SECC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_SECC_CMN_TLS == true){ execute(TC_SECC_CMN_VTB_ServiceDetail_007());};

    if (PIXIT_SECC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1){
        execute(TC_SECC_CMN_VTB_ServiceDetail_004());};

    if (PIXIT_SECC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1){
        execute(TC_SECC_CMN_VTB_ServiceDetail_006());};

    if (PIXIT_SECC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1){
        execute(TC_SECC_CMN_VTB_ServiceDetail_008());};

    // PaymentServiceSelection ++++++//
    if (true){ execute(TC_SECC_CMN_VTB_PaymentServiceSelection_004());};

    if (true){ execute(TC_SECC_CMN_VTB_PaymentServiceSelection_005());};

    if (true){ execute(TC_SECC_CMN_VTB_PaymentServiceSelection_006());};

    if (true){ execute(TC_SECC_CMN_VTB_PaymentServiceSelection_007());};

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_SECC_CMN_PaymentOption == externalPayment){
        execute(TC_SECC_CMN_VTB_PaymentServiceSelection_009());};

    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_SECC_CMN_PaymentOption == contract){
        execute(TC_SECC_CMN_VTB_PaymentServiceSelection_010());};

    // CertificateInstallation+++++//
    if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1){
        execute(TC_SECC_CMN_VTB_CertificateInstallation_002());};

    if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
        PICS_CMN_CMN_IdentificationMode == pnC and

```

```

PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateInstallation_003());};

if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateInstallation_005());};

if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateInstallation_006());};

if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateInstallation_007());};

if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateInstallation_008());};

// CertificateUpdate ++++++//
if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_002());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_003());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_005());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_006());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_007());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_008());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_009());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_010());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and

```



```

PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_011());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_012());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_013());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_014());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_015());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_016());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_017());};

// PaymentDetails ++++++//
if (PIXIT_SECC_CMN_CertificateHandling == none_ and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_PaymentDetails_004());};

if (PIXIT_SECC_CMN_CertificateHandling == none_ and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_PaymentDetails_005());};

if (PIXIT_SECC_CMN_CertificateHandling == none_ and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_PaymentDetails_007());};

if (PIXIT_SECC_CMN_CertificateHandling == none_ and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_PaymentDetails_008());};

if (PIXIT_SECC_CMN_CertificateHandling == none_ and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_PaymentDetails_009());};

if (PIXIT_SECC_CMN_CertificateHandling == none_ and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_PaymentDetails_010());};

// Authorization ++++++//
if (PICS_CMN_CMN_IdentificationMode == eIM){

```

```

        execute(TC_SECC_CMN_VTB_Authorization_003());

if (PICS_CMN_CMN_IdentificationMode == eIM){
    execute(TC_SECC_CMN_VTB_Authorization_005());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_004());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_006());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_007());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_010());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_011());

// ChargeParameterDiscovery ++++++//
if (true){ execute(TC_SECC_CMN_VTB_ChargeParameterDiscovery_001());

if (true){ execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_003());

if (true){ execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_004());

if (true){ execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_005());

// PowerDelivery Start ++++++//
if (true){ execute(TC_SECC_AC_VTB_PowerDelivery_006());

if (true){ execute(TC_SECC_AC_VTB_PowerDelivery_008());

if (true){ execute(TC_SECC_AC_VTB_PowerDelivery_010());

if (true){ execute(TC_SECC_AC_VTB_PowerDelivery_011());

if (true){ execute(TC_SECC_AC_VTB_PowerDelivery_014());

// ChargingStatus ++++++//
if (true){ execute(TC_SECC_AC_VTB_ChargingStatus_003());

if (true){ execute(TC_SECC_AC_VTB_ChargingStatus_004());

// MeteringReceipt ++++++//
if(PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_ReceiptRequired and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_AC_VTB_MeteringReceipt_002());

if(PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_ReceiptRequired and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_AC_VTB_MeteringReceipt_003());

if(PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_ReceiptRequired and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_AC_VTB_MeteringReceipt_004());

if(PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_ReceiptRequired and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){

```

```

        execute(TC_SECC_AC_VTB_MeteringReceipt_006());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_ReceiptRequired and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_AC_VTB_MeteringReceipt_007());

// PowerDelivery Stop ++++++
if (true){ execute(TC_SECC_AC_VTB_PowerDelivery_007());

if (true){ execute(TC_SECC_AC_VTB_PowerDelivery_009());

// SessionStop ++++++
if (true){ execute(TC_SECC_AC_VTB_SessionStop_002());

if (true){ execute(TC_SECC_AC_VTB_SessionStop_003());
}

/** Test Group 4: V2G timeouts */
if (PICS_CMN_CMN_V2gTimeouts) {

// SupportedAppProtocol ++++++
if (true){ execute(TC_SECC_CMN_VTB_SupportedAppProtocol_003());

if (true){ execute(TC_SECC_CMN_VTB_SupportedAppProtocol_005());

// SessionSetup ++++++
if (true){ execute(TC_SECC_CMN_VTB_SessionSetup_003());

if (true){ execute(TC_SECC_CMN_VTB_SessionSetup_004());

// ServiceDiscovery ++++++
if (true){ execute(TC_SECC_CMN_VTB_ServiceDiscovery_004());

// ServiceDetail ++++++
if (PIXIT_SECC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_SECC_CMN_TLS == true){ execute(TC_SECC_CMN_VTB_ServiceDetail_009());

if (PIXIT_SECC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_ServiceDetail_010());

// PaymentServiceSelection ++++++
if (true){ execute(TC_SECC_CMN_VTB_PaymentServiceSelection_008());

// CertificateInstallation ++++++
if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_CertificateInstallation_004());

// CertificateUpdate ++++++
if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_CertificateUpdate_004());

// PaymentDetails ++++++
if (PIXIT_SECC_CMN_CertificateHandling == none_ and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_PaymentDetails_006());

// Authorization ++++++
if (PICS_CMN_CMN_IdentificationMode == eIM){
    execute(TC_SECC_CMN_VTB_Authorization_008());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){

```

```

        execute(TC_SECC_CMN_VTB_Authorization_009());

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PICS_SECC_CMN_EIMDone == v2gAuthorization){
    execute(TC_SECC_CMN_VTB_Authorization_014());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_SECC_CMN_AuthorizationOngoing == ongoing){
    execute(TC_SECC_CMN_VTB_Authorization_015());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_SECC_CMN_AuthorizationOngoing == ongoing){
    execute(TC_SECC_CMN_VTB_Authorization_016());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_SECC_CMN_AuthorizationOngoing == ongoing){
    execute(TC_SECC_CMN_VTB_Authorization_017());

// ChargeParameterDiscovery ++++++//
if (true){ execute(TC_SECC_CMN_VTB_ChargeParameterDiscovery_002());

if (PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_014());

// PowerDelivery Start ++++++//
if (true){ execute(TC_SECC_AC_VTB_PowerDelivery_012());

// ChargingStatus ++++++//
if (true){ execute(TC_SECC_AC_VTB_ChargingStatus_005());

// MeteringReceipt ++++++//
if(PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_ReceiptRequired and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_AC_VTB_MeteringReceipt_005());

// PowerDelivery Stop ++++++//
if (true){ execute(TC_SECC_AC_VTB_PowerDelivery_013());

// SessionStop ++++++//
if (true){ execute(TC_SECC_AC_VTB_SessionStop_004());
    }
}
}

```

```

module AC_SECC_Charging_Control {

    import from Pics_15118_2 all;
    import from Pixit_15118_2 all;
    import from Timer_15118_2 all;
    import from Pics_15118 all;
    import from TestCases_SECC_V2GTPSDP all;
    import from TestCases_SECC_SDP all;
    import from TestCases_SECC_V2GTPSessionSetup all;
    import from TestCases_SECC_SessionSetup all;
    import from TestCases_SECC_SupportedAppProtocol all;
    import from TestCases_SECC_ServiceDiscovery all;
    import from TestCases_SECC_ServiceDetail all;
    import from TestCases_SECC_PaymentServiceSelection all;
    import from TestCases_SECC_CertificateInstallation all;
    import from TestCases_SECC_CertificateUpdate all;
    import from TestCases_SECC_PaymentDetails all;
    import from TestCases_SECC_Authorization all;
    import from TestCases_SECC_ServiceDetail all;
    import from TestCases_SECC_ChargeParameterDiscovery all;
    import from TestCases_SECC_PowerDelivery all;
    import from TestCases_SECC_ChargingStatus all;

```

```

import from TestCases_SECC_MeteringReceipt all;
import from TestCases_SECC_SessionStop all;
import from ComponentsAndPorts all;
import from Timer_15118 all;
import from Pixit_15118 all;

control {

    // V2GTP explicit and SDP ++++++//
    if (PICS_CMN_CMN_V2gtpSdp){ execute(TC_SECC_CMN_VTB_V2GTPSDP_001());};

    // SDP ++++++//
    if (PICS_CMN_CMN_IdentificationMode == eIM and not
        PIXIT_SECC_CMN_TLS) {execute(TC_SECC_CMN_VTB_SDP_001());};

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_SECC_CMN_TLS) {execute(TC_SECC_CMN_VTB_SDP_002());};

    if (PICS_CMN_CMN_IdentificationMode == pnC) {execute(TC_SECC_CMN_VTB_SDP_003());};

    if (PICS_CMN_CMN_IdentificationMode == eIM and not
        PIXIT_SECC_CMN_TLS) {
        execute(TC_SECC_CMN_VTB_SDP_007());};

    // SupportedAppProtocol ++++++//
    if (PICS_CMN_CMN_IdentificationMode == pnC or
        (PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_SECC_CMN_TLS)){
        execute(TC_SECC_CMN_VTB_SupportedAppProtocol_001());};

    if (PICS_CMN_CMN_IdentificationMode == eIM and not PIXIT_SECC_CMN_TLS){
        execute(TC_SECC_CMN_VTB_SupportedAppProtocol_004());};

    if (true){ execute(TC_SECC_CMN_VTB_SupportedAppProtocol_006());};

    // V2GTPSessionSetup ++++++//
    if (PICS_CMN_CMN_V2gtpV2g){ execute(TC_SECC_CMN_VTB_V2GTPSessionSetup_001());};

    // SessionSetup ++++++//
    if (true){ execute(TC_SECC_CMN_VTB_SessionSetup_001());};

    if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp){
        execute(TC_SECC_AC_VTB_SessionSetup_001());};

    if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp){
        execute(TC_SECC_AC_VTB_SessionSetup_002());};

    if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp){
        execute(TC_SECC_AC_VTB_SessionSetup_003());};

    if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp){
        execute(TC_SECC_AC_VTB_SessionSetup_004());};

    if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp){
        execute(TC_SECC_AC_VTB_SessionSetup_005());};

    if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp){
        execute(TC_SECC_AC_VTB_SessionSetup_006());};

    // ServiceDiscovery ++++++//
    if (true){ execute(TC_SECC_CMN_VTB_ServiceDiscovery_001());};

    if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp){
        execute(TC_SECC_AC_VTB_ServiceDiscovery_001());};

    if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp){
        execute(TC_SECC_AC_VTB_ServiceDiscovery_002());};

```

```

// ServiceDetail ++++++//
if (PIXIT_SECC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_SECC_CMN_TLS and
    PIXIT_SECC_CMN_InternetAccess == unknown){
    execute(TC_SECC_CMN_VTB_ServiceDetail_001());};

if (PIXIT_SECC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_SECC_CMN_InternetAccess == unknown){
    execute(TC_SECC_CMN_VTB_ServiceDetail_002());};

if (PIXIT_SECC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_SECC_CMN_TLS and
    PIXIT_SECC_CMN_InternetAccess == internetAccess){
    execute(TC_SECC_CMN_VTB_ServiceDetail_011());};

if (PIXIT_SECC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_SECC_CMN_InternetAccess == internetAccess){
    execute(TC_SECC_CMN_VTB_ServiceDetail_012());};

// PaymentServiceSelection ++++++//
if (PIXIT_SECC_CMN_VAS == none_){
    execute(TC_SECC_CMN_VTB_PaymentServiceSelection_001());};

if (PIXIT_SECC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_SECC_CMN_TLS){
    execute(TC_SECC_CMN_VTB_PaymentServiceSelection_002());};

if (PIXIT_SECC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_PaymentServiceSelection_003());};

// CertificateInstallation ++++++//
if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_CertificateInstallation_001());};

// CertificateUpdate ++++++//
if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_CertificateUpdate_001());};

// PaymentDetails ++++++//
if (PIXIT_SECC_CMN_CertificateHandling == none_ and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_PaymentDetails_001());};

if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_PaymentDetails_002());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){

```



```

        execute(TC_SECC_CMN_VTB_PaymentDetails_003());

// Authorization ++++++//
if (PICS_CMN_CMN_IdentificationMode == eIM){
    execute(TC_SECC_CMN_VTB_Authorization_001());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_002());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_012());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_013());

// Start Charging Loop ++++++//
// ChargeParameterDiscovery ++++++//
if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_001());

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_006());

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_008());

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_010());

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_012());

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_013());

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_015());

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_016());

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown and
    PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and
    PICS_CMN_CMN_CombinedTesting){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_007());

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff and
    PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and

```

```

PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_011());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == unknown and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_017());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == salesTariff and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_018());};

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
not PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == unknown and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_019());};

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
not PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == salesTariff and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_020());};

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == salesTariff and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_021());};

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == unknown and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_022());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
not PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == unknown and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_029());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
not PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == salesTariff and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_030());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == unknown and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_031());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == salesTariff and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_032());};

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
not PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == unknown and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_033());};

```

```

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff and
    PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
    PICS_CMN_CMN_CombinedTesting){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_034());};

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff and
    PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
    PICS_CMN_CMN_CombinedTesting){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_035());};

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown and
    PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
    PICS_CMN_CMN_CombinedTesting){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_036());};

// PowerDelivery Start ++++++//
if (PICS_SECC_CMN_EvStateInChargeProgress == C){
    execute(TC_SECC_AC_VTB_PowerDelivery_001());};

if (PICS_SECC_CMN_EvStateInChargeProgress == D){
    execute(TC_SECC_AC_VTB_PowerDelivery_002());};

// ChargingStatus ++++++//
if (PICS_CMN_CMN_IdentificationMode == eIM or
    (not PIXIT_SECC_CMN_ReceiptRequired and
     PICS_CMN_CMN_IdentificationMode == pnC) and
    PIXIT_SECC_CMN_MeterInfo == unknown){
    execute(TC_SECC_AC_VTB_ChargingStatus_001());};

if (PIXIT_SECC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_SECC_CMN_MeterInfo == unknown){
    execute(TC_SECC_AC_VTB_ChargingStatus_002());};

if (PICS_CMN_CMN_IdentificationMode == eIM or
    (not PIXIT_SECC_CMN_ReceiptRequired and
     PICS_CMN_CMN_IdentificationMode == pnC) and
    PIXIT_SECC_CMN_MeterInfo == meterInfo and
    PIXIT_CMN_CMN_MeterReading == unknown){
    execute(TC_SECC_AC_VTB_ChargingStatus_006());};

if (PIXIT_SECC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_SECC_CMN_MeterInfo == meterInfo and
    PIXIT_CMN_CMN_MeterReading == unknown){
    execute(TC_SECC_AC_VTB_ChargingStatus_007());};

if (PICS_CMN_CMN_IdentificationMode == eIM or
    (not PIXIT_SECC_CMN_ReceiptRequired and
     PICS_CMN_CMN_IdentificationMode == pnC) and
    PIXIT_SECC_CMN_MeterInfo == meterInfo and
    PIXIT_CMN_CMN_MeterReading == meterReading){
    execute(TC_SECC_AC_VTB_ChargingStatus_008());};

if (PIXIT_SECC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_SECC_CMN_MeterInfo == meterInfo and
    PIXIT_CMN_CMN_MeterReading == meterReading){
    execute(TC_SECC_AC_VTB_ChargingStatus_009());};

// MeteringReceipt ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_ReceiptRequired and
    PICS_CMN_CMN_PkiName == PKI_1 and

```

```

PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_AC_VTB_MeteringReceipt_001());};

// Renegotiation ++++++//

// PowerDelivery Renegotiate ++++++//
if (PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown ) {
    execute(TC_SECC_AC_VTB_PowerDelivery_003());};

// ChargeParameterDiscovery Renegotiate ++++++//
if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown and
    PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_002());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff and
    PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_009());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown and
    PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_023());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff and
    PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_024());};

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown and
    PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_025());};

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff and
    PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_026());};

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff and
    PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_027());};

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown and
    PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown){
    execute(TC_SECC_AC_VTB_ChargeParameterDiscovery_028());};

// PowerDelivery Start after Renegotiate ++++++//
if (PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown){
    execute(TC_SECC_AC_VTB_PowerDelivery_004());};

// PowerDelivery Stop ++++++//
if (true){ execute(TC_SECC_AC_VTB_PowerDelivery_005());};

```

```

// End Charging Loop ++++++//

// SessionStop ++++++//
if (true){ execute(TC_SECC_AC_VTB_SessionStop_001());}
}
}

```

B.1.2 DC specific control parts

```

module DC_SECC_Error_Control {

import from Pics_15118_2 all;
import from Pixit_15118_2 all;
import from Pics_15118 all;
import from TestCases_SECC_V2GTPSDP all;
import from TestCases_SECC_SDP all;
import from TestCases_SECC_V2GTPSessionSetup all;
import from TestCases_SECC_SessionSetup all;
import from TestCases_SECC_SupportedAppProtocol all;
import from TestCases_SECC_ServiceDiscovery all;
import from TestCases_SECC_ServiceDetail all;
import from TestCases_SECC_PaymentServiceSelection all;
import from TestCases_SECC_CertificateInstallation all;
import from TestCases_SECC_CertificateUpdate all;
import from TestCases_SECC_PaymentDetails all;
import from TestCases_SECC_Authorization all;
import from TestCases_SECC_ServiceDetail all;
import from TestCases_SECC_ChargeParameterDiscovery all;
import from TestCases_SECC_CableCheck all;
import from TestCases_SECC_PreCharge all;
import from TestCases_SECC_PowerDelivery all;
import from TestCases_SECC_CurrentDemand all;
import from TestCases_SECC_ChargingStatus all;
import from TestCases_SECC_MeteringReceipt all;
import from TestCases_SECC_WeldingDetection all;
import from TestCases_SECC_SessionStop all;
import from ComponentsAndPorts all;

control {

/** Test Group 1: Invalid V2GTP data fields */
if (PICS_CMN_CMN_InvalidV2gtpDataFields) {

// V2GTP explicit and SDP ++++++//
if (PICS_CMN_CMN_V2gtpSdp){ execute(TC_SECC_CMN_VTB_V2GTPSDP_002());}

if (PICS_CMN_CMN_V2gtpSdp){ execute(TC_SECC_CMN_VTB_V2GTPSDP_003());}

if (PICS_CMN_CMN_V2gtpSdp){ execute(TC_SECC_CMN_VTB_V2GTPSDP_004());}

if (PICS_CMN_CMN_V2gtpSdp){ execute(TC_SECC_CMN_VTB_V2GTPSDP_005());}

// V2GTP explicit and V2G (SessionSetup) ++++++//
if (PICS_CMN_CMN_V2gtpV2g){ execute(TC_SECC_CMN_VTB_V2GTPSessionSetup_002());}

if (PICS_CMN_CMN_V2gtpV2g){ execute(TC_SECC_CMN_VTB_V2GTPSessionSetup_003());}

if (PICS_CMN_CMN_V2gtpV2g){ execute(TC_SECC_CMN_VTB_V2GTPSessionSetup_004());}

if (PICS_CMN_CMN_V2gtpV2g){ execute(TC_SECC_CMN_VTB_V2GTPSessionSetup_005());}

}

/** Test Group 2: SDP timeouts and invalid SDP data fields */
if (PICS_CMN_CMN_SDPTimeoutsAndInvalidData) {

// SDP ++++++//
if (PICS_CMN_CMN_IdentificationMode == eIM and not
PIXIT_SECC_CMN_TLS) {execute(TC_SECC_CMN_VTB_SDP_004());}

if (PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_SECC_CMN_TLS) {execute(TC_SECC_CMN_VTB_SDP_005());}

if (PICS_CMN_CMN_IdentificationMode == pnC) {execute(TC_SECC_CMN_VTB_SDP_006());}

}

}
}

```

```

/** Test Group 3: V2G FAILED ResponseCodes */

if (PICS_CMN_CMN_FailedResponseCodes) {

    // SupportedAppProtocol ++++++//
    if (true){ execute(TC_SECC_CMN_VTB_SupportedAppProtocol_002());};

    // SessionSetup ++++++//
    if (true){ execute(TC_SECC_CMN_VTB_SessionSetup_002());};

    // ServiceDiscovery ++++++//
    if (true){ execute(TC_SECC_CMN_VTB_ServiceDiscovery_002());};

    if (true){ execute(TC_SECC_CMN_VTB_ServiceDiscovery_003());};

    // ServiceDetail ++++++//
    if (PIXIT_SECC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_SECC_CMN_TLS == true){ execute(TC_SECC_CMN_VTB_ServiceDetail_003());};

    if (PIXIT_SECC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_SECC_CMN_TLS == true){ execute(TC_SECC_CMN_VTB_ServiceDetail_005());};

    if (PIXIT_SECC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_SECC_CMN_TLS == true){ execute(TC_SECC_CMN_VTB_ServiceDetail_007());};

    if (PIXIT_SECC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1){
        execute(TC_SECC_CMN_VTB_ServiceDetail_004());};

    if (PIXIT_SECC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1){
        execute(TC_SECC_CMN_VTB_ServiceDetail_006());};

    if (PIXIT_SECC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1){
        execute(TC_SECC_CMN_VTB_ServiceDetail_008());};

    // PaymentServiceSelection ++++++//
    if (true){ execute(TC_SECC_CMN_VTB_PaymentServiceSelection_004());};

    if (true){ execute(TC_SECC_CMN_VTB_PaymentServiceSelection_005());};

    if (true){ execute(TC_SECC_CMN_VTB_PaymentServiceSelection_006());};

    if (true){ execute(TC_SECC_CMN_VTB_PaymentServiceSelection_007());};

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_SECC_CMN_PaymentOption == externalPayment){
        execute(TC_SECC_CMN_VTB_PaymentServiceSelection_009());};

    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_SECC_CMN_PaymentOption == contract){
        execute(TC_SECC_CMN_VTB_PaymentServiceSelection_010());};

    // CertificateInstallation ++++++//
    if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1){
        execute(TC_SECC_CMN_VTB_CertificateInstallation_002());};

    if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1){
        execute(TC_SECC_CMN_VTB_CertificateInstallation_003());};

    if (PIXIT_SECC_CMN_CertificateHandling == certInstall and

```



```

PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateInstallation_005());};

if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateInstallation_006());};

if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateInstallation_007());};

if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateInstallation_008());};

// CertificateUpdate ++++++//
if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_002());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_003());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_005());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_006());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_007());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_008());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_009());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_010());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_011());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and

```

```

PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_012());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_013());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_014());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_015());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_016());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateUpdate_017());};

// PaymentDetails ++++++//
if (PIXIT_SECC_CMN_CertificateHandling == none_ and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_PaymentDetails_004());};

if (PIXIT_SECC_CMN_CertificateHandling == none_ and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_PaymentDetails_005());};

if (PIXIT_SECC_CMN_CertificateHandling == none_ and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_PaymentDetails_007());};

if (PIXIT_SECC_CMN_CertificateHandling == none_ and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_PaymentDetails_008());};

if (PIXIT_SECC_CMN_CertificateHandling == none_ and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_PaymentDetails_009());};

if (PIXIT_SECC_CMN_CertificateHandling == none_ and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_PaymentDetails_010());};

// Authorization ++++++//
if (PICS_CMN_CMN_IdentificationMode == eIM){
execute(TC_SECC_CMN_VTB_Authorization_003());};

if (PICS_CMN_CMN_IdentificationMode == eIM){
execute(TC_SECC_CMN_VTB_Authorization_005());};

```

```

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_004());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_006());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_007());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_010());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_011());};

// ChargeParameterDiscovery ++++++//
if (true){ execute(TC_SECC_CMN_VTB_ChargeParameterDiscovery_001());};

if (true){ execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_002());};

if (true){ execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_003());};

if (true){ execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_004());};

// CableCheck ++++++//
if (true){execute(TC_SECC_DC_VTB_CableCheck_003());};

if (true){execute(TC_SECC_DC_VTB_CableCheck_004());};

if (){execute(TC_SECC_DC_VTB_CableCheck_008());};

// PreCharge ++++++//
if (true){execute(TC_SECC_DC_VTB_PreCharge_002());};

if (true){execute(TC_SECC_DC_VTB_PreCharge_003());};

// PowerDelivery Start ++++++//
if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_003());};

if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_005());};

if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_007());};

if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_008());};

if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_013());};

// CurrentDemand ++++++//
if (true){ execute(TC_SECC_DC_VTB_CurrentDemand_003());};

if (true){ execute(TC_SECC_DC_VTB_CurrentDemand_004());};

// MeteringReceipt ++++++//
if(PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_ReceiptRequired and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_DC_VTB_MeteringReceipt_002());};

if(PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_ReceiptRequired and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_DC_VTB_MeteringReceipt_003());};

if(PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_ReceiptRequired and
    PICS_CMN_CMN_PkiName == PKI_1 and

```

```

PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_DC_VTB_MeteringReceipt_004());};

if(PICS_CMN_CMN_IdentificationMode == pnC and
PIXIT_SECC_CMN_ReceiptRequired and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_DC_VTB_MeteringReceipt_006());};

if(PICS_CMN_CMN_IdentificationMode == pnC and
PIXIT_SECC_CMN_ReceiptRequired and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_DC_VTB_MeteringReceipt_007());};

// PowerDelivery Stop ++++++//
if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_004());};

if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_006());};

// WeldingDetection ++++++//
if (PICS_SECC_DC_WeldingDetection){ execute(TC_SECC_DC_VTB_WeldingDetection_002());};

if (PICS_SECC_DC_WeldingDetection){ execute(TC_SECC_DC_VTB_WeldingDetection_003());};

if (PICS_SECC_DC_WeldingDetection){ execute(TC_SECC_DC_VTB_WeldingDetection_005());};

// SessionStop ++++++//
if (not PICS_SECC_DC_WeldingDetection){ execute(TC_SECC_DC_VTB_SessionStop_003());};

if (not PICS_SECC_DC_WeldingDetection){ execute(TC_SECC_DC_VTB_SessionStop_004());};

if (not PICS_SECC_DC_WeldingDetection){ execute(TC_SECC_DC_VTB_SessionStop_006());};

if (PICS_SECC_DC_WeldingDetection){ execute(TC_SECC_DC_VTB_SessionStop_007());};

if (PICS_SECC_DC_WeldingDetection){ execute(TC_SECC_DC_VTB_SessionStop_008());};
}

/** Test Group 4: V2G timeouts */
if (PICS_CMN_CMN_V2gTimeouts){

// SupportedAppProtocol ++++++//
if (true){ execute(TC_SECC_CMN_VTB_SupportedAppProtocol_003());};

if (true){ execute(TC_SECC_CMN_VTB_SupportedAppProtocol_005());};

// SessionSetup ++++++//
if (true){ execute(TC_SECC_CMN_VTB_SessionSetup_003());};

if (true){ execute(TC_SECC_CMN_VTB_SessionSetup_004());};

// ServiceDiscovery ++++++//
if (true){ execute(TC_SECC_CMN_VTB_ServiceDiscovery_004());};

// ServiceDetail ++++++//
if (PIXIT_SECC_CMN_VAS == serviceDetail and
PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_SECC_CMN_TLS == true){ execute(TC_SECC_CMN_VTB_ServiceDetail_009());};

if (PIXIT_SECC_CMN_VAS == serviceDetail and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_ServiceDetail_010());};

// PaymentServiceSelection ++++++//
if (true){ execute(TC_SECC_CMN_VTB_PaymentServiceSelection_008());};

// CertificateInstallation ++++++//
if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_CMN_VTB_CertificateInstallation_004());};

// CertificateUpdate ++++++//

```

```

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_CertificateUpdate_004());};

// PaymentDetails ++++++
if (PIXIT_SECC_CMN_CertificateHandling == none_ and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_PaymentDetails_006());};

// Authorization ++++++
if (PICS_CMN_CMN_IdentificationMode == eIM){
    execute(TC_SECC_CMN_VTB_Authorization_008());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_009());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PICS_SECC_CMN_EIMDone == v2gAuthorization){
    execute(TC_SECC_CMN_VTB_Authorization_014());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_SECC_CMN_AuthorizationOngoing == ongoing){
    execute(TC_SECC_CMN_VTB_Authorization_015());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_SECC_CMN_AuthorizationOngoing == ongoing){
    execute(TC_SECC_CMN_VTB_Authorization_016());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_SECC_CMN_AuthorizationOngoing == ongoing){
    execute(TC_SECC_CMN_VTB_Authorization_017());};

// ChargeParameterDiscovery ++++++
if (true){ execute(TC_SECC_CMN_VTB_ChargeParameterDiscovery_002());};

if (PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_014());};

// CableCheck ++++++
if (true){ execute(TC_SECC_DC_VTB_CableCheck_005());};

// PreCharge ++++++
if (true){ execute(TC_SECC_DC_VTB_PreCharge_004());};

// PowerDelivery Start ++++++
if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_009());};

// CurrentDemand ++++++
if (true){ execute(TC_SECC_DC_VTB_CurrentDemand_005());};

// MeteringReceipt ++++++
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_ReceiptRequired and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_DC_VTB_MeteringReceipt_005());};

// PowerDelivery Stop ++++++
if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_010());};

// WeldingDetection ++++++
if (PICS_SECC_DC_WeldingDetection){ execute(TC_SECC_DC_VTB_WeldingDetection_004());};

// SessionStop ++++++
if (not PICS_SECC_DC_WeldingDetection){ execute(TC_SECC_DC_VTB_SessionStop_005());};

```

```

    if (PICS_SECC_DC_WeldingDetection){ execute(TC_SECC_DC_VTB_SessionStop_009()); }
  }
}

module DC_SECC_Charging_Control {

  import from PICS_15118_2 all;
  import from Pixit_15118_2 all;
  import from Timer_15118_2 all;
  import from PICS_15118 all;
  import from TestCases_SECC_V2GTPSDP all;
  import from TestCases_SECC_SDP all;
  import from TestCases_SECC_SessionSetup all;
  import from TestCases_SECC_V2GTPSessionSetup all;
  import from TestCases_SECC_SupportedAppProtocol all;
  import from TestCases_SECC_ServiceDiscovery all;
  import from TestCases_SECC_ServiceDetail all;
  import from TestCases_SECC_PaymentServiceSelection all;
  import from TestCases_SECC_CertificateInstallation all;
  import from TestCases_SECC_CertificateUpdate all;
  import from TestCases_SECC_PaymentDetails all;
  import from TestCases_SECC_Authorization all;
  import from TestCases_SECC_ServiceDetail all;
  import from TestCases_SECC_ChargeParameterDiscovery all;
  import from TestCases_SECC_CableCheck all;
  import from TestCases_SECC_PreCharge all;
  import from TestCases_SECC_PowerDelivery all;
  import from TestCases_SECC_CurrentDemand all;
  import from TestCases_SECC_ChargingStatus all;
  import from TestCases_SECC_MeteringReceipt all;
  import from TestCases_SECC_WeldingDetection all;
  import from TestCases_SECC_SessionStop all;
  import from ComponentsAndPorts all;
  import from Timer_15118 all;
  import from Pixit_15118 all;

  control {

    // V2GTP explicit and SDP ++++++
    if (PICS_CMN_CMN_V2gtpSdp){ execute(TC_SECC_CMN_VTB_V2GTPSDP_001()); }

    // SDP ++++++
    if (PICS_CMN_CMN_IdentificationMode == eIM and not
        PIXIT_SECC_CMN_TLS) {execute(TC_SECC_CMN_VTB_SDP_001()); }

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_SECC_CMN_TLS) {execute(TC_SECC_CMN_VTB_SDP_002()); }

    if (PICS_CMN_CMN_IdentificationMode == pnC) {execute(TC_SECC_CMN_VTB_SDP_003()); }

    if (PICS_CMN_CMN_IdentificationMode == eIM and not
        PIXIT_SECC_CMN_TLS) {
      execute(TC_SECC_CMN_VTB_SDP_007()); }

    // SupportedAppProtocol ++++++
    if (PICS_CMN_CMN_IdentificationMode == pnC or
        (PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_SECC_CMN_TLS)){
      execute(TC_SECC_CMN_VTB_SupportedAppProtocol_001()); }

    if (PICS_CMN_CMN_IdentificationMode == eIM and not PIXIT_SECC_CMN_TLS){
      execute(TC_SECC_CMN_VTB_SupportedAppProtocol_004()); }

    if (true){ execute(TC_SECC_CMN_VTB_SupportedAppProtocol_006()); }

    // V2GTPSessionSetup ++++++
    if (PICS_CMN_CMN_V2gtpV2g){ execute(TC_SECC_CMN_VTB_V2GTPSessionSetup_001()); }

    // SessionSetup ++++++
    if (true){ execute(TC_SECC_CMN_VTB_SessionSetup_001()); }
  }
}

```



```

if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp){
    execute(TC_SECC_DC_VTB_SessionSetup_001());};

if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp){
    execute(TC_SECC_DC_VTB_SessionSetup_002());};

if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp){
    execute(TC_SECC_DC_VTB_SessionSetup_003());};

if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp){
    execute(TC_SECC_DC_VTB_SessionSetup_004());};

if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp){
    execute(TC_SECC_DC_VTB_SessionSetup_005());};

if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp){
    execute(TC_SECC_DC_VTB_SessionSetup_006());};

// ServiceDiscovery ++++++
if (true){ execute(TC_SECC_CMN_VTB_ServiceDiscovery_001());};

if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp){
    execute(TC_SECC_DC_VTB_ServiceDiscovery_001());};

if (PICS_SECC_CMN_Pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp){
    execute(TC_SECC_DC_VTB_ServiceDiscovery_002());};

// ServiceDetail ++++++
if (PIXIT_SECC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_SECC_CMN_TLS and
    PIXIT_SECC_CMN_InternetAccess == unknown){
    execute(TC_SECC_CMN_VTB_ServiceDetail_001());};

if (PIXIT_SECC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_SECC_CMN_InternetAccess == unknown){
    execute(TC_SECC_CMN_VTB_ServiceDetail_002());};

if (PIXIT_SECC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_SECC_CMN_TLS and
    PIXIT_SECC_CMN_InternetAccess == internetAccess){
    execute(TC_SECC_CMN_VTB_ServiceDetail_011());};

if (PIXIT_SECC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_SECC_CMN_InternetAccess == internetAccess){
    execute(TC_SECC_CMN_VTB_ServiceDetail_012());};

// PaymentServiceSelection ++++++
if (PIXIT_SECC_CMN_VAS == none){
    execute(TC_SECC_CMN_VTB_PaymentServiceSelection_001());};

if (PIXIT_SECC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_SECC_CMN_TLS){
    execute(TC_SECC_CMN_VTB_PaymentServiceSelection_002());};

if (PIXIT_SECC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_PaymentServiceSelection_003());};

```

```

// CertificateInstallation ++++++//
if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_CertificateInstallation_001());};

// CertificateUpdate ++++++//
if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_CertificateUpdate_001());};

// PaymentDetails ++++++//
if (PIXIT_SECC_CMN_CertificateHandling == none_ and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_PaymentDetails_001());};

if (PIXIT_SECC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_PaymentDetails_002());};

if (PIXIT_SECC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_PaymentDetails_003());};

// Authorization ++++++//
if (PICS_CMN_CMN_IdentificationMode == eIM){
    execute(TC_SECC_CMN_VTB_Authorization_001());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_002());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_012());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_SECC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1){
    execute(TC_SECC_CMN_VTB_Authorization_013());};

// Start Charging Loop ++++++//

// ChargeParameterDiscovery ++++++//
if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_001());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_006());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_008());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and

```

```

PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == salesTariff){
execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_010());}

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
not PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == unknown){
execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_012());}

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
not PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == salesTariff){
execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_013());}

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == salesTariff){
execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_015());}

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == unknown){
execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_016());}

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
not PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == unknown and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_007());}

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
not PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == salesTariff and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_011());}

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == unknown and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_017());}

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == salesTariff and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_018());}

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
not PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == unknown and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_019());}

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
not PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == salesTariff and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_020());}

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == salesTariff and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_021());}

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == unknown and
PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp and
PICS_CMN_CMN_CombinedTesting){
execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_022());}

```

```

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown and
    PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
    PICS_CMN_CMN_CombinedTesting){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_029());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff and
    PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
    PICS_CMN_CMN_CombinedTesting){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_030());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown and
    PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
    PICS_CMN_CMN_CombinedTesting){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_031());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff and
    PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
    PICS_CMN_CMN_CombinedTesting){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_032());};

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown and
    PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
    PICS_CMN_CMN_CombinedTesting){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_033());};

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff and
    PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
    PICS_CMN_CMN_CombinedTesting){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_034());};

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff and
    PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
    PICS_CMN_CMN_CombinedTesting){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_035());};

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown and
    PICS_SECC_CMN_Pause and PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp and
    PICS_CMN_CMN_CombinedTesting){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_036());};

// CableCheck ++++++
if (PICS_SECC_CMN_EvStateInChargeProgress == C and
    PIXIT_SECC_DC_CableCheckOngoing == unknown){
    execute(TC_SECC_DC_VTB_CableCheck_001());};

if (PICS_SECC_CMN_EvStateInChargeProgress == D and
    PIXIT_SECC_DC_CableCheckOngoing == unknown){
    execute(TC_SECC_DC_VTB_CableCheck_002());};

if (PICS_SECC_CMN_EvStateInChargeProgress == C and
    PIXIT_SECC_DC_CableCheckOngoing == ongoing){
    execute(TC_SECC_DC_VTB_CableCheck_006());};

if (PICS_SECC_CMN_EvStateInChargeProgress == D and
    PIXIT_SECC_DC_CableCheckOngoing == ongoing){
    execute(TC_SECC_DC_VTB_CableCheck_007());};

// PreCharge ++++++
if (true){ execute(TC_SECC_DC_VTB_PreCharge_001());};

```

```

// PowerDelivery Start ++++++//
if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_001());};

// CurrentDemand ++++++//
if (PICS_CMN_CMN_IdentificationMode == eIM or
(not PIXIT_SECC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC) and
PIXIT_SECC_CMN_MeterInfo == unknown){
execute(TC_SECC_DC_VTB_CurrentDemand_001());};

if (PIXIT_SECC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PIXIT_SECC_CMN_MeterInfo == unknown){
execute(TC_SECC_DC_VTB_CurrentDemand_002());};

if (PICS_CMN_CMN_IdentificationMode == eIM or
(not PIXIT_SECC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC) and
PIXIT_SECC_CMN_MeterInfo == meterInfo and
PIXIT_CMN_CMN_MeterReading == unknown){
execute(TC_SECC_DC_VTB_CurrentDemand_006());};

if (PIXIT_SECC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PIXIT_SECC_CMN_MeterInfo == meterInfo and
PIXIT_CMN_CMN_MeterReading == unknown){
execute(TC_SECC_DC_VTB_CurrentDemand_007());};

if (PICS_CMN_CMN_IdentificationMode == eIM or
(not PIXIT_SECC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC) and
PIXIT_SECC_CMN_MeterInfo == meterInfo and
PIXIT_CMN_CMN_MeterReading == meterReading){
execute(TC_SECC_DC_VTB_CurrentDemand_008());};

if (PIXIT_SECC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PIXIT_SECC_CMN_MeterInfo == meterInfo and
PIXIT_CMN_CMN_MeterReading == meterReading){
execute(TC_SECC_DC_VTB_CurrentDemand_009());};

// MeteringReceipt ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
PIXIT_SECC_CMN_ReceiptRequired and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1){
execute(TC_SECC_DC_VTB_MeteringReceipt_001());};

// Renegotiation ++++++//
// PowerDelivery Renegotiate ++++++//
if (PICS_CMN_CMN_Renegotiation and
PIXIT_CMN_CMN_Renegotiation == unknown) {
execute(TC_SECC_DC_VTB_PowerDelivery_011());};

// ChargeParameterDiscovery Renegotiate ++++++//
if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
not PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == unknown and
PICS_CMN_CMN_Renegotiation and
PIXIT_CMN_CMN_Renegotiation == unknown){
execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_005());};

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
not PICS_SECC_CMN_DepartureTime and
PIXIT_SECC_CMN_SalesTariff == salesTariff and
PICS_CMN_CMN_Renegotiation and
PIXIT_CMN_CMN_Renegotiation == unknown){

```

```

        execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_009());

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown and
    PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_023());

if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff and
    PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_024());

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown and
    PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_025());

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    not PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff and
    PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_026());

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == salesTariff and
    PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_027());

if (PICS_SECC_CMN_MaxEntriesSAScheduleTuple and
    PICS_SECC_CMN_DepartureTime and
    PIXIT_SECC_CMN_SalesTariff == unknown and
    PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown){
    execute(TC_SECC_DC_VTB_ChargeParameterDiscovery_028());

// PowerDelivery Start after Renegotiate ++++++//
if (PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown ){
    execute(TC_SECC_DC_VTB_PowerDelivery_012());

// PowerDelivery Stop ++++++//
if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_002());

if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_014());

if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_015());

if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_016());

if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_017());

if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_018());

if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_019());

if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_020());

if (true){ execute(TC_SECC_DC_VTB_PowerDelivery_021());

// End Charging Loop ++++++//

// WeldingDetection ++++++//
if (PICS_SECC_DC_WeldingDetection){ execute(TC_SECC_DC_VTB_WeldingDetection_001());

// SessionStop ++++++//
if (not PICS_SECC_DC_WeldingDetection){ execute(TC_SECC_DC_VTB_SessionStop_001());

```



```

    if (PICS_SECC_DC_WeldingDetection){ execute(TC_SECC_DC_VTB_SessionStop_002()); }
}

```

B.2 EVCC control parts

B.2.1 AC specific control parts

```

module AC_EVCC_Error_Control {

    import from PICS_15118_2 all;
    import from Pixit_15118_2 all;
    import from PICS_15118 all;
    import from TestCases_EVCC_V2GTPSDP all;
    import from TestCases_EVCC_SDP all;
    import from TestCases_EVCC_SupportedAppProtocol all;
    import from TestCases_EVCC_SessionSetup all;
    import from TestCases_EVCC_V2GTPSessionSetup all;
    import from TestCases_EVCC_ServiceDiscovery all;
    import from TestCases_EVCC_ServiceDetail all;
    import from TestCases_EVCC_ServiceDetailAndPaymentSelection all;
    import from TestCases_EVCC_CertificateHandlingAndPaymentDetails all;
    import from TestCases_EVCC_CertificateInstallation all;
    import from TestCases_EVCC_CertificateUpdate all;
    import from TestCases_EVCC_Authorization all;
    import from TestCases_EVCC_ChargeParameterDiscovery all;
    import from TestCases_EVCC_PowerDelivery all;
    import from TestCases_EVCC_ChargingStatus all;
    import from TestCases_EVCC_ChargingLoopAcOrPowerDelivery all;
    import from TestCases_EVCC_SessionStop all;
    import from TestCases_EVCC_ChargeParameterDiscovery all;
    import from TestCases_EVCC_PowerDelivery all;
    import from TestCases_EVCC_ChargingStatus all;
    import from TestCases_EVCC_MeteringReceipt all;
    import from TestCases_EVCC_SessionStop all;
    import from ComponentsAndPorts all;

    control {

        /** Test Group 1: Invalid V2GTP data fields */

        if (PICS_CMN_CMN_InvalidV2gtpDataFields) {

            // V2GTP explicit and SDP ++++++
            if (PICS_CMN_CMN_V2gtpSdp){ execute(TC_EVCC_CMN_VTB_V2GTPSDP_002()); }

            if (PICS_CMN_CMN_V2gtpSdp){ execute(TC_EVCC_CMN_VTB_V2GTPSDP_003()); }

            if (PICS_CMN_CMN_V2gtpSdp){ execute(TC_EVCC_CMN_VTB_V2GTPSDP_004()); }

            if (PICS_CMN_CMN_V2gtpSdp){ execute(TC_EVCC_CMN_VTB_V2GTPSDP_005()); }

            // V2GTP explicit and V2G (SessionSetup) ++++++
            if (PICS_CMN_CMN_V2gtpV2g) { execute(TC_EVCC_CMN_VTB_V2GTPSessionSetup_002()); }

            if (PICS_CMN_CMN_V2gtpV2g) { execute(TC_EVCC_CMN_VTB_V2GTPSessionSetup_003()); }

            if (PICS_CMN_CMN_V2gtpV2g) { execute(TC_EVCC_CMN_VTB_V2GTPSessionSetup_004()); }

            if (PICS_CMN_CMN_V2gtpV2g) { execute(TC_EVCC_CMN_VTB_V2GTPSessionSetup_005()); }

        }

        /** Test Group 2: SDP timeouts and invalid SDP data fields */

        if (PICS_CMN_CMN_SDPTimeoutsAndInvalidData) {

            // SDP ++++++
            if (true) { execute(TC_EVCC_CMN_VTB_SDP_005()); }

            if (true) { execute(TC_EVCC_CMN_VTB_SDP_007()); }

        }

        /** Test Group 3: V2G FAILED ResponseCodes */
    }
}

```

```

if (PICS_CMN_CMN_FailedResponseCodes) {

    // SupportedAppProtocol ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_SupportedAppProtocol_002());};

    // SessionSetup ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_SessionSetup_002());};

    if (true) { execute(TC_EVCC_CMN_VTB_SessionSetup_003());};

    // ServiceDiscovery ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_002());};

    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_003());};

    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_004());};

    // ServiceDetail ++++++//
    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_ServiceDetail_003());};

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_ServiceDetail_005());};

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_ServiceDetail_007());};

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_ServiceDetail_009());};

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_ServiceDetail_004());};

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_ServiceDetail_006());};

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_ServiceDetail_008());};

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_ServiceDetail_010());};

    // PaymentServiceSelection ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_003());};

    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_004());};

    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_005());};

    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_006());};

    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_007());};

    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_008());};

    // CertificateInstallation ++++++//
    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_CertificateHandling == certInstall and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_CertificateInstallation_002());};

    if (PICS_CMN_CMN_IdentificationMode == pnC and

```

```

PIXIT_EVCC_CMN_CertificateHandling == certInstall and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_003());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_004());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_005());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_006());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_007());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_008());
}

// CertificateUpdate ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_002());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_003());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_004());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_005());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_006());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_007());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_008());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and

```

```

PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_009());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_010());
}

// PaymentDetails ++++++
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_002());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_003());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_004());
}

// Authorization ++++++
if (true) { execute(TC_EVCC_CMN_VTB_Authorization_003());
}

if (true) { execute(TC_EVCC_CMN_VTB_Authorization_004());
}

if (true) { execute(TC_EVCC_CMN_VTB_Authorization_005());
}

if (true) { execute(TC_EVCC_CMN_VTB_Authorization_006());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_Authorization_007());
}

// ChargeParameterDiscovery ++++++
if (true) { execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_002());
}

if (true) { execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_003());
}

if (true) { execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_004());
}

if (true) { execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_005());
}

if (true) { execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_006());
}

// PowerDelivery Start ++++++
if (true) { execute(TC_EVCC_AC_VTB_PowerDelivery_002());
}

if (true) { execute(TC_EVCC_AC_VTB_PowerDelivery_003());
}

if (true) { execute(TC_EVCC_AC_VTB_PowerDelivery_004());
}

if (true) { execute(TC_EVCC_AC_VTB_PowerDelivery_005());
}

if (true) { execute(TC_EVCC_AC_VTB_PowerDelivery_006());
}

if (true) { execute(TC_EVCC_AC_VTB_PowerDelivery_007());
}

if (true) { execute(TC_EVCC_AC_VTB_PowerDelivery_008());
}

// ChargingStatus ++++++
if (true) { execute(TC_EVCC_AC_VTB_ChargingStatus_002());
}

if (true) { execute(TC_EVCC_AC_VTB_ChargingStatus_003());
}

if (true) { execute(TC_EVCC_AC_VTB_ChargingStatus_004());
}

// MeteringReceipt ++++++
if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {

```

```

        execute(TC_EVCC_AC_VTB_MeteringReceipt_002());

    if (PICS_EVCC_CMN_ReceiptRequired and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_AC_VTB_MeteringReceipt_003());
    }

    if (PICS_EVCC_CMN_ReceiptRequired and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_AC_VTB_MeteringReceipt_004());
    }

    if (PICS_EVCC_CMN_ReceiptRequired and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_AC_VTB_MeteringReceipt_005());
    }

    if (PICS_EVCC_CMN_ReceiptRequired and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_AC_VTB_MeteringReceipt_006());
    }

    // PowerDelivery Stop ++++++//
    if (true) { execute(TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_003()); }

    if (true) { execute(TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_004()); }

    if (true) { execute(TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_005()); }

    if (true) { execute(TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_006()); }

    // SessionStop ++++++//
    if (true) { execute(TC_EVCC_AC_VTB_SessionStop_002()); }

    if (true) { execute(TC_EVCC_AC_VTB_SessionStop_003()); }

    if (true) { execute(TC_EVCC_AC_VTB_SessionStop_004()); }
}

/** Test Group 4: Invalid V2G messages */

if (PICS_EVCC_CMN_InvalidV2GMessages) {

    // SessionSetup ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_SessionSetup_005()); }

    // ServiceDiscovery ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_008()); }

    // ServiceDetail ++++++//
    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_ServiceDetail_013()); }

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_ServiceDetail_014());
    }

    // PaymentServiceSelection ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_010()); }

    // CertificateInstallation ++++++//
    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_CertificateHandling == certInstall and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_CertificateInstallation_010());
    }

    // CertificateUpdate ++++++//
    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {

```

```

        execute(TC_EVCC_CMN_VTB_CertificateUpdate_012());

// PaymentDetails ++++++
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_006());
}

// Authorization ++++++
if (true) { execute(TC_EVCC_CMN_VTB_Authorization_012()); }

// ChargeParameterDiscovery ++++++
if (true) { execute(TC_EVCC_CMN_VTB_ChargeParameterDiscovery_002()); }

// PowerDelivery Start ++++++
if (true) { execute(TC_EVCC_AC_VTB_PowerDelivery_010()); }

// ChargingStatus ++++++
if (true) { execute(TC_EVCC_AC_VTB_ChargingStatus_008()); }

// MeteringReceipt ++++++
if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_AC_VTB_MeteringReceipt_010());
}

// PowerDelivery Stop ++++++
if (true) { execute(TC_EVCC_AC_VTB_ChargingLoopAcOnPowerDelivery_010()); }

// SessionStop ++++++
if (true) { execute(TC_EVCC_AC_VTB_SessionStop_006()); }
}

/** Test Group 5: V2G timeouts */

if (PICS_CMN_CMN_V2gTimeouts) {

// SupportedAppProtocol ++++++
if (true) { execute(TC_EVCC_CMN_VTB_SupportedAppProtocol_003()); }

// SessionSetup ++++++
if (true) { execute(TC_EVCC_CMN_VTB_SessionSetup_004()); }

// ServiceDiscovery ++++++
if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_005()); }

// ServiceDetail ++++++
if (PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_ServiceDetail_011()); }

if (PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_ServiceDetail_012());
}

// PaymentServiceSelection ++++++
if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009()); }

// CertificateInstallation ++++++
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_009());
}

// CertificateUpdate ++++++
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_011());
}

// PaymentDetails ++++++
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and

```



```

    PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005());

// Authorization ++++++//
if (true) { execute(TC_EVCC_CMN_VTB_Authorization_008());}

if (true) { execute(TC_EVCC_CMN_VTB_Authorization_009());}

// ChargeParameterDiscovery ++++++//
if (true) { execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_007());}

if (true) { execute(TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001());}

// PowerDelivery Start ++++++//
if (true) { execute(TC_EVCC_AC_VTB_PowerDelivery_009());}

// ChargingStatus ++++++//
if (true) { execute(TC_EVCC_AC_VTB_ChargingStatus_005());}

// MeteringReceipt ++++++//
if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_AC_VTB_MeteringReceipt_007());}

// PowerDelivery Stop ++++++//
if (true) { execute(TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007());}

// SessionStop ++++++//
if (true) { execute(TC_EVCC_AC_VTB_SessionStop_005());}
}

/** Test Group 6: Invalid states and duty cycles */
if (PICS_EVCC_CMN_InvalidStatesAndDutyCycles) {

// SessionSetup ++++++//
if (true) { execute(TC_EVCC_CMN_VTB_SessionSetup_006());}

if (true) { execute(TC_EVCC_CMN_VTB_SessionSetup_007());}

// ServiceDiscovery ++++++//
if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_006());}

if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_007());}

// ServiceDetail ++++++//
if (PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == tls) {
    execute(TC_EVCC_CMN_VTB_ServiceDetail_015());}

if (PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_ServiceDetail_016());}

if (PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == tls) {
    execute(TC_EVCC_CMN_VTB_ServiceDetail_017());}

if (PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_ServiceDetail_018());}

// PaymentServiceSelection ++++++//
if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_011());}

if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_012());}

// CertificateInstallation ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and

```

```

        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
            execute(TC_EVCC_CMN_VTB_CertificateInstallation_011());
        }

    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_CertificateHandling == certInstall and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_CertificateInstallation_012());
    }

    // CertificateUpdate ++++++
    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_CertificateUpdate_013());
    }

    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_CertificateUpdate_014());
    }

    // PaymentDetails ++++++
    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_007());
    }

    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_008());
    }

    // Authorization ++++++
    if (true) { execute(TC_EVCC_CMN_VTB_Authorization_013()); }

    if (true) { execute(TC_EVCC_CMN_VTB_Authorization_014()); }

    // ChargeParameterDiscovery ++++++
    if (true) { execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_009()); }

    if (true) { execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_010()); }

    // PowerDelivery Start ++++++
    if (true) { execute(TC_EVCC_AC_VTB_PowerDelivery_011()); }

    if (true) { execute(TC_EVCC_AC_VTB_PowerDelivery_012()); }

    // ChargingStatus ++++++
    if (true) { execute(TC_EVCC_AC_VTB_ChargingStatus_006()); }

    if (true) { execute(TC_EVCC_AC_VTB_ChargingStatus_007()); }

    // MeteringReceipt ++++++
    if (PICS_EVCC_CMN_ReceiptRequired and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_AC_VTB_MeteringReceipt_011());
    }

    if (PICS_EVCC_CMN_ReceiptRequired and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_AC_VTB_MeteringReceipt_012());
    }

    // PowerDelivery Stop ++++++
    if (true) { execute(TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_011()); }

    if (true) { execute(TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_012()); }

}
}
}

```

```

module AC_EVCC_Charging_Control {

```

```

import from Pics_15118_2 all;
import from Pixit_15118_2 all;
import from Pics_15118 all;
import from Pixit_15118 all;
import from TestCases_EVCC_V2GTPSDP all;
import from TestCases_EVCC_SDP all;
import from TestCases_EVCC_SupportedAppProtocol all;
import from TestCases_EVCC_V2GTPSessionSetup all;
import from TestCases_EVCC_SessionSetup all;
import from TestCases_EVCC_ServiceDiscovery all;
import from TestCases_EVCC_ServiceDetailAndPaymentSelection all;
import from TestCases_EVCC_PaymentServiceSelection all;
import from TestCases_EVCC_CertificateHandlingAndPaymentDetails all;
import from TestCases_EVCC_CertificateInstallation all;
import from TestCases_EVCC_CertificateUpdate all;
import from TestCases_EVCC_PaymentDetails all;
import from TestCases_EVCC_Authorization all;
import from TestCases_EVCC_ChargeParameterDiscovery all;
import from TestCases_EVCC_PowerDelivery all;
import from TestCases_EVCC_ChargingStatus all;
import from TestCases_EVCC_ChargingLoopAcOrPowerDelivery all;
import from TestCases_EVCC_ServiceDetail all;
import from TestCases_EVCC_ChargeParameterDiscovery all;
import from TestCases_EVCC_PowerDelivery all;
import from TestCases_EVCC_ChargingStatus all;
import from TestCases_EVCC_MeteringReceipt all;
import from TestCases_EVCC_SessionStop all;
import from ComponentsAndPorts all;
import from Timer_15118 all;
import from Timer_15118_2 all;

control {

    // V2GTP explicit and SDP ++++++
    if (PICS_CMN_CMN_V2gtpSdp) { execute(TC_EVCC_CMN_VTB_V2GTPSDP_001()); }

    // SDP ++++++
    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == none_) { execute(TC_EVCC_CMN_VTB_SDP_001()); }

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_SDP_002()); }

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == unknown) { execute(TC_EVCC_CMN_VTB_SDP_003()); }

    if (PICS_CMN_CMN_IdentificationMode == pnC) { execute(TC_EVCC_CMN_VTB_SDP_004()); }

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == none_ and
        PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_AC_VTB_SDP_001());
    }

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == tls and
        PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_AC_VTB_SDP_002());
    }

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == unknown and
        PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_AC_VTB_SDP_003());
    }

    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_AC_VTB_SDP_004());
    }

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == none_ and
        PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_AC_VTB_SDP_005());
    }
}

```

```

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == tls and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_SDP_006());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == unknown and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_SDP_007());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_SDP_008());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == none_ and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_SDP_009());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == tls and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_SDP_010());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == unknown and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_SDP_011());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_SDP_012());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == none_ and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_SDP_013());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == tls and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_SDP_014());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == unknown and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_SDP_015());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_SDP_016());};

if (true) { execute(TC_EVCC_CMN_VTB_SDP_006());};

// SupportedAppProtocol ++++++//
if ((PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == tls) or
    PICS_CMN_CMN_IdentificationMode == pnC) {
    execute(TC_EVCC_CMN_VTB_SupportedAppProtocol_001());};

```

```

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == none_) { execute(TC_EVCC_CMN_VTB_SupportedAppProtocol_004());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == unknown) { execute(TC_EVCC_CMN_VTB_SupportedAppProtocol_005());};

// V2GTPSessionSetup ++++++//
if (PICS_CMN_CMN_V2gtpV2g) { execute(TC_EVCC_CMN_VTB_V2GTPSessionSetup_001());};

// SessionSetup ++++++//
if (not PICS_CMN_CMN_CombinedTesting) { execute(TC_EVCC_CMN_VTB_SessionSetup_001());};

if (PICS_CMN_CMN_CombinedTesting) { execute(TC_EVCC_CMN_VTB_SessionSetup_008());};

if (PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_SessionSetup_001());};

if (PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_SessionSetup_002());};

if (PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_SessionSetup_003());};

if (PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_SessionSetup_004());};

// ServiceDiscovery ++++++//
if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_001());};

// ServiceDetail ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) { execute(TC_EVCC_CMN_VTB_ServiceDetail_001());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_ServiceDetail_002());};

// PaymentServiceSelection ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_PaymentServiceSelection_001());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_PaymentServiceSelection_002());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == none_ and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_PaymentServiceSelection_003());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == none_) {
    execute(TC_EVCC_CMN_VTB_PaymentServiceSelection_004());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == none_) {
    execute(TC_EVCC_AC_VTB_PaymentServiceSelection_001());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == unknown and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {

```

```

        execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001());

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_VAS == unknown) {
        execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002());

    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1 and
        PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_AC_VTB_PaymentServiceSelection_002());

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_VAS == serviceDetail and
        PIXIT_EVCC_CMN_TLS == tls and
        PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_AC_VTB_PaymentServiceSelection_003());

    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_VAS == none_ and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1 and
        PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_AC_VTB_PaymentServiceSelection_004());

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_VAS == none_ and
        PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_AC_VTB_PaymentServiceSelection_005());

    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_VAS == unknown and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1 and
        PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001());

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_VAS == unknown and
        PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002());

    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1 and
        PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_AC_VTB_PaymentServiceSelection_006());

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_VAS == serviceDetail and
        PIXIT_EVCC_CMN_TLS == tls and
        PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_AC_VTB_PaymentServiceSelection_007());

    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_VAS == none_ and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1 and
        PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_AC_VTB_PaymentServiceSelection_008());

    if (PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_VAS == none_ and
        PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
        PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_AC_VTB_PaymentServiceSelection_009());

    if (PICS_CMN_CMN_IdentificationMode == pnC and

```



```

PIXIT_EVCC_CMN_VAS == unknown and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_EVCC_CMN_VAS == unknown and
PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
PIXIT_EVCC_CMN_VAS == serviceDetail and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
PICS_CMN_CMN_CombinedTesting and
par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_AC_VTB_PaymentServiceSelection_010());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_EVCC_CMN_VAS == serviceDetail and
PIXIT_EVCC_CMN_TLS == tls and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
PICS_CMN_CMN_CombinedTesting and
par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_AC_VTB_PaymentServiceSelection_011());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
PIXIT_EVCC_CMN_VAS == none_ and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
PICS_CMN_CMN_CombinedTesting and
par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_AC_VTB_PaymentServiceSelection_012());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_EVCC_CMN_VAS == none_ and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
PICS_CMN_CMN_CombinedTesting and
par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_AC_VTB_PaymentServiceSelection_013());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
PIXIT_EVCC_CMN_VAS == unknown and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
PICS_CMN_CMN_CombinedTesting and
par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_EVCC_CMN_VAS == unknown and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
PICS_CMN_CMN_CombinedTesting and
par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
PIXIT_EVCC_CMN_VAS == serviceDetail and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
PICS_CMN_CMN_CombinedTesting and
par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_AC_VTB_PaymentServiceSelection_014());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_EVCC_CMN_VAS == serviceDetail and
PIXIT_EVCC_CMN_TLS == tls and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
PICS_CMN_CMN_CombinedTesting and
par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_AC_VTB_PaymentServiceSelection_015());};

```

```

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == none_ and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_PaymentServiceSelection_016());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == none_ and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_PaymentServiceSelection_017());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == unknown and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == unknown and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008());};

// CertificateInstallation ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_001());};

// CertificateUpdate ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_001());};

// PaymentDetails ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_PaymentDetails_001());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_PaymentDetails_002());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == none_ and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_PaymentDetails_003());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == unknown and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001());};

// Authorization ++++++//

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and

```

```

    PIXIT_CMN_CMN_PkiName == PKI_1) { execute(TC_EVCC_CMN_VTB_Authorization_001());};

if (PICS_CMN_CMN_IdentificationMode == eIM) { execute(TC_EVCC_CMN_VTB_Authorization_002());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_Authorization_010());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_Authorization_011());};

// Start Charging Loop ++++++
// ChargeParameterDiscovery ++++++
if (PIXIT_EVCC_CMN_DepartureTime == unknown) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_001());};

if (PIXIT_EVCC_CMN_DepartureTime == none_) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_012());};

if (PIXIT_EVCC_CMN_DepartureTime == unknown and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_015());};

if (PIXIT_EVCC_CMN_DepartureTime == none_ and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_016());};

if (PIXIT_EVCC_CMN_DepartureTime == departureTime and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_017());};

if (PIXIT_EVCC_CMN_DepartureTime == unknown and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_018());};

if (PIXIT_EVCC_CMN_DepartureTime == none_ and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_019());};

if (PIXIT_EVCC_CMN_DepartureTime == departureTime and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_020());};

if (PIXIT_EVCC_CMN_DepartureTime == unknown and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_021());};

if (PIXIT_EVCC_CMN_DepartureTime == none_ and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_022());};

if (PIXIT_EVCC_CMN_DepartureTime == departureTime and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_023());};

if (PIXIT_EVCC_CMN_DepartureTime == unknown and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_024());};

```

```

if (PIXIT_EVCC_CMN_DepartureTime == none_ and
    PICS_EVCC_CMN_PmaxScheduleWithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_025());};

if (PIXIT_EVCC_CMN_DepartureTime == departureTime and
    PICS_EVCC_CMN_PmaxScheduleWithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_026());};

// PowerDelivery Start ++++++//
if (PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == unknown and
    PIXIT_EVCC_CMN_ChargingProfileOptimization == unknown) {
    execute(TC_EVCC_AC_VTB_PowerDelivery_001());};

if (PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == maxNumberOfPhasesInUse and
    PIXIT_EVCC_CMN_ChargingProfileOptimization == unknown) {
    execute(TC_EVCC_AC_VTB_PowerDelivery_015());};

if (PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == unknown and
    PIXIT_EVCC_CMN_ChargingProfileOptimization == optimization and
    PICS_CMN_EVCC_SalesTariff) {
    execute(TC_EVCC_AC_VTB_PowerDelivery_018());};

if (PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == maxNumberOfPhasesInUse and
    PIXIT_EVCC_CMN_ChargingProfileOptimization == optimization and
    PICS_CMN_EVCC_SalesTariff) {
    execute(TC_EVCC_AC_VTB_PowerDelivery_019());};

if (PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == unknown and
    PIXIT_EVCC_CMN_ChargingProfileOptimization == none_ and
    PICS_CMN_EVCC_SalesTariff) {
    execute(TC_EVCC_AC_VTB_PowerDelivery_020());};

if (PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == maxNumberOfPhasesInUse and
    PIXIT_EVCC_CMN_ChargingProfileOptimization == none_ and
    PICS_CMN_EVCC_SalesTariff) {
    execute(TC_EVCC_AC_VTB_PowerDelivery_021());};

// ChargingStatus ++++++//
if (true) {execute(TC_EVCC_AC_VTB_ChargingStatus_001());};

// MeteringReceipt ++++++//
if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_MeterReading == unknown and
    PIXIT_EVCC_CMN_SAScheduleTupleIDMR == unknown) {
    execute(TC_EVCC_AC_VTB_MeteringReceipt_001());};

if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PIXIT_CMN_CMN_MeterReading == unknown and
    PIXIT_EVCC_CMN_SAScheduleTupleIDMR == unknown) {
    execute(TC_EVCC_AC_VTB_MeteringReceipt_008());};

if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PIXIT_CMN_CMN_MeterReading == unknown and
    PIXIT_EVCC_CMN_SAScheduleTupleIDMR == unknown) {
    execute(TC_EVCC_AC_VTB_MeteringReceipt_009());};

if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and

```

```

    PIXIT_CMN_CMN_MeterReading == meterReading and
    PIXIT_EVCC_CMN_SAScheduleTupleIDMR == unknown) {
        execute(TC_EVCC_AC_VTB_MeteringReceipt_013());
    }

    if (PICS_EVCC_CMN_ReceiptRequired and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1 and
        PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
        PIXIT_CMN_CMN_MeterReading == meterReading and
        PIXIT_EVCC_CMN_SAScheduleTupleIDMR == unknown) {
        execute(TC_EVCC_AC_VTB_MeteringReceipt_014());
    }

    if (PICS_EVCC_CMN_ReceiptRequired and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1 and
        PIXIT_EVCC_CMN_CertificateHandling == certInstall and
        PIXIT_CMN_CMN_MeterReading == meterReading and
        PIXIT_EVCC_CMN_SAScheduleTupleIDMR == unknown) {
        execute(TC_EVCC_AC_VTB_MeteringReceipt_015());
    }

    if (PICS_EVCC_CMN_ReceiptRequired and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_MeterReading == unknown and
        PIXIT_EVCC_CMN_SAScheduleTupleIDMR == sAScheduleTupleID) {
        execute(TC_EVCC_AC_VTB_MeteringReceipt_016());
    }

    if (PICS_EVCC_CMN_ReceiptRequired and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1 and
        PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
        PIXIT_CMN_CMN_MeterReading == unknown and
        PIXIT_EVCC_CMN_SAScheduleTupleIDMR == sAScheduleTupleID) {
        execute(TC_EVCC_AC_VTB_MeteringReceipt_017());
    }

    if (PICS_EVCC_CMN_ReceiptRequired and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1 and
        PIXIT_EVCC_CMN_CertificateHandling == certInstall and
        PIXIT_CMN_CMN_MeterReading == unknown and
        PIXIT_EVCC_CMN_SAScheduleTupleIDMR == sAScheduleTupleID) {
        execute(TC_EVCC_AC_VTB_MeteringReceipt_018());
    }

    if (PICS_EVCC_CMN_ReceiptRequired and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_MeterReading == meterReading and
        PIXIT_EVCC_CMN_SAScheduleTupleIDMR == sAScheduleTupleID) {
        execute(TC_EVCC_AC_VTB_MeteringReceipt_019());
    }

    if (PICS_EVCC_CMN_ReceiptRequired and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1 and
        PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
        PIXIT_CMN_CMN_MeterReading == meterReading and
        PIXIT_EVCC_CMN_SAScheduleTupleIDMR == sAScheduleTupleID) {
        execute(TC_EVCC_AC_VTB_MeteringReceipt_020());
    }

    if (PICS_EVCC_CMN_ReceiptRequired and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1 and
        PIXIT_EVCC_CMN_CertificateHandling == certInstall and
        PIXIT_CMN_CMN_MeterReading == meterReading and
        PIXIT_EVCC_CMN_SAScheduleTupleIDMR == sAScheduleTupleID) {
        execute(TC_EVCC_AC_VTB_MeteringReceipt_021());
    }

    // PowerDelivery Stop
    if (PICS_CMN_CMN_IdentificationMode == eIM or
        (not PICS_EVCC_CMN_ReceiptRequired and
         PICS_CMN_CMN_IdentificationMode == pnC)) {

```

```

        execute(TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001());

if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002());
}

// PowerDelivery Renegotiate ++++++//
if (PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown and
    PICS_CMN_CMN_IdentificationMode == eIM or
    (not PICS_EVCC_CMN_ReceiptRequired and
     PICS_CMN_CMN_IdentificationMode == pnC)) {
    execute(TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008());
}

if (PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown and
    PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009());
}

if (not PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == renegotiation and
    PICS_CMN_CMN_IdentificationMode == eIM or
    (not PICS_EVCC_CMN_ReceiptRequired and
     PICS_CMN_CMN_IdentificationMode == pnC)) {
    execute(TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013());
}

if (not PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == renegotiation and
    PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014());
}

// ChargeParameter after Renegotiation ++++++//
if (PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown and
    PIXIT_EVCC_CMN_DepartureTime == unknown) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_008());
}

if (not PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == renegotiation and
    PIXIT_EVCC_CMN_DepartureTime == unknown) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_011());
}

if (PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown and
    PIXIT_EVCC_CMN_DepartureTime == none_) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_013());
}

if (not PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == renegotiation and
    PIXIT_EVCC_CMN_DepartureTime == none_) {
    execute(TC_EVCC_AC_VTB_ChargeParameterDiscovery_014());
}

// PowerDelivery Start after Renegotiation ++++++//
if (PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown and
    PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == unknown and
    PIXIT_EVCC_CMN_ChargingProfileOptimization == unknown and
    PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
    execute(TC_EVCC_AC_VTB_PowerDelivery_013());
}

if (not PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == renegotiation and
    PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == unknown and
    PIXIT_EVCC_CMN_ChargingProfileOptimization == unknown and
    PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
    execute(TC_EVCC_AC_VTB_PowerDelivery_014());
}

if (PICS_CMN_CMN_Renegotiation and

```



```

    PIXIT_CMN_CMN_Renegotiation == unknown and
    PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == maxNumberOfPhasesInUse and
    PIXIT_EVCC_CMN_ChargingProfileOptimization == unknown and
    PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
        execute(TC_EVCC_AC_VTB_PowerDelivery_016());
    }

    if (not PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == renegotiation and
        PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == maxNumberOfPhasesInUse and
        PIXIT_EVCC_CMN_ChargingProfileOptimization == unknown and
        PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
        execute(TC_EVCC_AC_VTB_PowerDelivery_017());
    }

    if (PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == unknown and
        PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == unknown and
        PIXIT_EVCC_CMN_ChargingProfileOptimization == optimization and
        PICS_CMN_EVCC_SalesTariff and
        PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
        execute(TC_EVCC_AC_VTB_PowerDelivery_022());
    }

    if (not PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == renegotiation and
        PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == unknown and
        PIXIT_EVCC_CMN_ChargingProfileOptimization == optimization and
        PICS_CMN_EVCC_SalesTariff and
        PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
        execute(TC_EVCC_AC_VTB_PowerDelivery_023());
    }

    if (PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == unknown and
        PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == maxNumberOfPhasesInUse and
        PIXIT_EVCC_CMN_ChargingProfileOptimization == optimization and
        PICS_CMN_EVCC_SalesTariff and
        PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
        execute(TC_EVCC_AC_VTB_PowerDelivery_024());
    }

    if (not PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == renegotiation and
        PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == maxNumberOfPhasesInUse and
        PIXIT_EVCC_CMN_ChargingProfileOptimization == optimization and
        PICS_CMN_EVCC_SalesTariff and
        PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
        execute(TC_EVCC_AC_VTB_PowerDelivery_025());
    }

    if (PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == unknown and
        PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == unknown and
        PIXIT_EVCC_CMN_ChargingProfileOptimization == none_ and
        PICS_CMN_EVCC_SalesTariff and
        PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
        execute(TC_EVCC_AC_VTB_PowerDelivery_026());
    }

    if (not PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == renegotiation and
        PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == unknown and
        PIXIT_EVCC_CMN_ChargingProfileOptimization == none_ and
        PICS_CMN_EVCC_SalesTariff and
        PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
        execute(TC_EVCC_AC_VTB_PowerDelivery_027());
    }

    if (PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == unknown and
        PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == maxNumberOfPhasesInUse and
        PIXIT_EVCC_CMN_ChargingProfileOptimization == none_ and
        PICS_CMN_EVCC_SalesTariff and
        PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
        execute(TC_EVCC_AC_VTB_PowerDelivery_028());
    }

    if (not PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == renegotiation and
        PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == maxNumberOfPhasesInUse and
        PIXIT_EVCC_CMN_ChargingProfileOptimization == none_ and
        PICS_CMN_EVCC_SalesTariff and
        PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
        execute(TC_EVCC_AC_VTB_PowerDelivery_029());
    }

    if (PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == unknown and

```

```

    PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == unknown and
    PIXIT_EVCC_CMN_ChargingProfileOptimization == unknown and
    PIXIT_EVCC_CMN_NewSAScheduleTupleID == newSAScheduleTupleID) {
        execute(TC_EVCC_AC_VTB_PowerDelivery_030());};

if (not PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == renegotiation and
    PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == unknown and
    PIXIT_EVCC_CMN_ChargingProfileOptimization == unknown and
    PIXIT_EVCC_CMN_NewSAScheduleTupleID == newSAScheduleTupleID) {
    execute(TC_EVCC_AC_VTB_PowerDelivery_031());};

if (PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == unknown and
    PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == maxNumberOfPhasesInUse and
    PIXIT_EVCC_CMN_ChargingProfileOptimization == unknown and
    PIXIT_EVCC_CMN_NewSAScheduleTupleID == newSAScheduleTupleID) {
    execute(TC_EVCC_AC_VTB_PowerDelivery_032());};

if (not PICS_CMN_CMN_Renegotiation and
    PIXIT_CMN_CMN_Renegotiation == renegotiation and
    PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == maxNumberOfPhasesInUse and
    PIXIT_EVCC_CMN_ChargingProfileOptimization == unknown and
    PIXIT_EVCC_CMN_NewSAScheduleTupleID == newSAScheduleTupleID) {
    execute(TC_EVCC_AC_VTB_PowerDelivery_033());};

// End Charging Loop ++++++//

// SessionStop ++++++//
if ((PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == tls) or
    PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_Pause == unknown) {
    execute(TC_EVCC_AC_VTB_SessionStop_001());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == none_ and
    PIXIT_EVCC_CMN_Pause == unknown) {
    execute(TC_EVCC_AC_VTB_SessionStop_007());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == unknown and
    PIXIT_EVCC_CMN_Pause == unknown) {
    execute(TC_EVCC_AC_VTB_SessionStop_008());};

if ((PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == tls) or
    PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_Pause == none_) {
    execute(TC_EVCC_AC_VTB_SessionStop_009());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == none_ and
    PIXIT_EVCC_CMN_Pause == none_) {
    execute(TC_EVCC_AC_VTB_SessionStop_010());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == unknown and
    PIXIT_EVCC_CMN_Pause == none_) {
    execute(TC_EVCC_AC_VTB_SessionStop_011());};

if ((PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == tls) or
    PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_Pause == pause) {
    execute(TC_EVCC_AC_VTB_SessionStop_012());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == none_ and
    PIXIT_EVCC_CMN_Pause == pause) {
    execute(TC_EVCC_AC_VTB_SessionStop_013());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == unknown and
    PIXIT_EVCC_CMN_Pause == pause) {
    execute(TC_EVCC_AC_VTB_SessionStop_014());};

if ((PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == tls) or

```

```

PICS_CMN_CMN_IdentificationMode == pnC and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge) {
    execute(TC_EVCC_AC_VTB_SessionStop_015());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == none_ and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge) {
    execute(TC_EVCC_AC_VTB_SessionStop_016());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == unknown and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge) {
    execute(TC_EVCC_AC_VTB_SessionStop_017());};

if ((PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == tls) or
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge) {
    execute(TC_EVCC_AC_VTB_SessionStop_018());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == none_ and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge) {
    execute(TC_EVCC_AC_VTB_SessionStop_019());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == unknown and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge) {
    execute(TC_EVCC_AC_VTB_SessionStop_020());};
}

```

B.2.2 DC specific control parts

```

module DC_EVCC_Error_Control {

    import from Pics_15118_2 all;
    import from Pixit_15118_2 all;
    import from Pics_15118 all;
    import from TestCases_EVCC_V2GTPSDP all;
    import from TestCases_EVCC_SDP all;
    import from TestCases_EVCC_SupportedAppProtocol all;
    import from TestCases_EVCC_V2GTPSessionSetup all;
    import from TestCases_EVCC_SessionSetup all;
    import from TestCases_EVCC_ServiceDiscovery all;
    import from TestCases_EVCC_ServiceDetail all;
    import from TestCases_EVCC_ServiceDetailAndPaymentSelection all;
    import from TestCases_EVCC_CertificateHandlingAndPaymentDetails all;
    import from TestCases_EVCC_CertificateInstallation all;
    import from TestCases_EVCC_CertificateUpdate all;
    import from TestCases_EVCC_Authorization all;
    import from TestCases_EVCC_ChargeParameterDiscovery all;
    import from TestCases_EVCC_PowerDelivery all;
    import from TestCases_EVCC_MeteringReceipt all;
    import from TestCases_EVCC_CableCheck all;
    import from TestCases_EVCC_PreCharge all;
    import from TestCases_EVCC_PreChargeOrPowerDelivery all;
    import from TestCases_EVCC_CurrentDemand all;
    import from TestCases_EVCC_ChargingLoopDcOrPowerDelivery all;
    import from TestCases_EVCC_WeldingDetectionOrSessionStop all;
    import from TestCases_EVCC_WeldingDetection all;
    import from TestCases_EVCC_SessionStop all;
    import from ComponentsAndPorts all;

    control {

        /** Test Group 1: Invalid V2GTP data fields */

        if (PICS_CMN_CMN_InvalidV2gtpDataFields) {

            // V2GTP explicit and SDP ++++++//
            if (PICS_CMN_CMN_V2gtpSdp) { execute(TC_EVCC_CMN_VTB_V2GTPSDP_002());};

            if (PICS_CMN_CMN_V2gtpSdp) { execute(TC_EVCC_CMN_VTB_V2GTPSDP_003());};

            if (PICS_CMN_CMN_V2gtpSdp) { execute(TC_EVCC_CMN_VTB_V2GTPSDP_004());};

            if (PICS_CMN_CMN_V2gtpSdp) { execute(TC_EVCC_CMN_VTB_V2GTPSDP_005());};
        }
    }
}

```

```

// V2GTP explicit and V2G (SessionSetup) ++++++//
if (PICS_CMN_CMN_V2gtpV2g) { execute(TC_EVCC_CMN_VTB_V2GTPSessionSetup_002());};

if (PICS_CMN_CMN_V2gtpV2g) { execute(TC_EVCC_CMN_VTB_V2GTPSessionSetup_003());};

if (PICS_CMN_CMN_V2gtpV2g) { execute(TC_EVCC_CMN_VTB_V2GTPSessionSetup_004());};

if (PICS_CMN_CMN_V2gtpV2g) { execute(TC_EVCC_CMN_VTB_V2GTPSessionSetup_005());};
}

/** Test Group 2: SDP timeouts and invalid SDP data fields */
if (PICS_CMN_CMN_SDPTimeoutsAndInvalidData) {

    // SDP ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_SDP_005());};

    if (true) { execute(TC_EVCC_CMN_VTB_SDP_007());};
}

/** Test Group 3: V2G FAILED ResponseCodes and invalid V2G messages */
if (PICS_CMN_CMN_FailedResponseCodes) {

    // SupportedAppProtocol ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_SupportedAppProtocol_002());};

    // SessionSetup ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_SessionSetup_002());};

    if (true) { execute(TC_EVCC_CMN_VTB_SessionSetup_003());};

    // ServiceDiscovery ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_002());};

    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_003());};

    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_004());};

    // ServiceDetail ++++++//
    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_ServiceDetail_003());};

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_ServiceDetail_005());};

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_ServiceDetail_007());};

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_ServiceDetail_009());};

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_ServiceDetail_004());};

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_ServiceDetail_006());};

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_ServiceDetail_008());};

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and

```

```

PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_ServiceDetail_010());};

// PaymentServiceSelection ++++++//
if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_003());};

if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_004());};

if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_005());};

if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_006());};

if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_007());};

if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_008());};

// CertificateInstallation ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_002());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_003());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_004());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_005());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_006());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_007());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_008());};

// CertificateUpdate ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_002());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_003());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_004());};

if (PICS_CMN_CMN_IdentificationMode == pnC and

```

```

PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
execute(TC_EVCC_CMN_VTB_CertificateUpdate_005());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
execute(TC_EVCC_CMN_VTB_CertificateUpdate_006());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
execute(TC_EVCC_CMN_VTB_CertificateUpdate_007());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
execute(TC_EVCC_CMN_VTB_CertificateUpdate_008());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
execute(TC_EVCC_CMN_VTB_CertificateUpdate_009());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
execute(TC_EVCC_CMN_VTB_CertificateUpdate_010());};

// PaymentDetails ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_002());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_003());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_004());};

// Authorization ++++++//
if (true) { execute(TC_EVCC_CMN_VTB_Authorization_003());};

if (true) { execute(TC_EVCC_CMN_VTB_Authorization_004());};

if (true) { execute(TC_EVCC_CMN_VTB_Authorization_005());};

if (true) { execute(TC_EVCC_CMN_VTB_Authorization_006());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
execute(TC_EVCC_CMN_VTB_Authorization_007());};

// ChargeParameterDiscovery ++++++//
if (true) { execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_002());};

if (true) { execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_003());};

if (true) { execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_004());};

if (true) { execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_005());};

if (true) { execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_006());};

// CableCheck ++++++//
if (true) { execute(TC_EVCC_DC_VTB_CableCheck_002());};

```



```

if (true) { execute(TC_EVCC_DC_VTB_CableCheck_003());};

if (true) { execute(TC_EVCC_DC_VTB_CableCheck_004());};

// PreCharge ++++++
if (true) { execute(TC_EVCC_DC_VTB_PreCharge_002());};

if (true) { execute(TC_EVCC_DC_VTB_PreCharge_003());};

if (true) { execute(TC_EVCC_DC_VTB_PreCharge_004());};

// PowerDelivery Start ++++++
if (true) { execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_002());};

if (true) { execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_003());};

if (true) { execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_004());};

if (true) { execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_005());};

if (true) { execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_006());};

if (true) { execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_007());};

if (true) { execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_008());};

// CurrentDemand ++++++
if (true) { execute(TC_EVCC_DC_VTB_CurrentDemand_002());};

if (true) { execute(TC_EVCC_DC_VTB_CurrentDemand_003());};

if (true) { execute(TC_EVCC_DC_VTB_CurrentDemand_004());};

// MeteringReceipt ++++++
if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_DC_VTB_MeteringReceipt_002());};

if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_DC_VTB_MeteringReceipt_003());};

if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_DC_VTB_MeteringReceipt_004());};

if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_DC_VTB_MeteringReceipt_005());};

if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_DC_VTB_MeteringReceipt_006());};

// PowerDelivery Stop ++++++
if (true) { execute(TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_003());};

if (true) { execute(TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_004());};

if (true) { execute(TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_005());};

if (true) { execute(TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_006());};

// WeldingDetection ++++++
if (PIXIT_EVCC_DC_WeldingDetection == weldingDetection) {
    execute(TC_EVCC_DC_VTB_WeldingDetection_002());};

if (PIXIT_EVCC_DC_WeldingDetection == weldingDetection) {
    execute(TC_EVCC_DC_VTB_WeldingDetection_003());};

```

```

if (PIXIT_EVCC_DC_WeldingDetection == weldingDetection) {
    execute(TC_EVCC_DC_VTB_WeldingDetection_004());

    // SessionStop ++++++//
    if (true) { execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_002());}

    if (true) { execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_003());}

    if (true) { execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_004());}
}

/** Test Group 4: Invalid V2G messages */

if (PICS_EVCC_CMN_InvalidV2GMessages) {

    // SessionSetup ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_SessionSetup_005());}

    // ServiceDiscovery ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_008());}

    // ServiceDetail ++++++//
    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_ServiceDetail_013());}

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_ServiceDetail_014());
    }

    // PaymentServiceSelection ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_010());}

    // CertificateInstallation ++++++//
    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_CertificateHandling == certInstall and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_CertificateInstallation_010());
    }

    // CertificateUpdate ++++++//
    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_CertificateUpdate_012());
    }

    // PaymentDetails ++++++//
    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_006());
    }

    // Authorization ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_Authorization_012());}

    // ChargeParameterDiscovery ++++++//
    if (true) { execute(TC_EVCC_CMN_VTB_ChargeParameterDiscovery_002());}

    // CableCheck ++++++//
    if (true) { execute(TC_EVCC_DC_VTB_CableCheck_007());}

    // PreCharge ++++++//
    if (true) { execute(TC_EVCC_DC_VTB_PreCharge_007());}

    // PowerDelivery Start ++++++//
    if (true) { execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_010());}

    // CurrentDemand ++++++//
    if (true) { execute(TC_EVCC_DC_VTB_CurrentDemand_008());}

    // MeteringReceipt ++++++//
    if (PICS_EVCC_CMN_ReceiptRequired and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {

```

```

        execute(TC_EVCC_DC_VTB_MeteringReceipt_010());

// PowerDelivery Stop ++++++
if (true) { execute(TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_010());}

// WeldingDetection ++++++
if (PIXIT_EVCC_DC_WeldingDetection == weldingDetection) {
    execute(TC_EVCC_DC_VTB_WeldingDetection_006());}

// SessionStop ++++++
if (true) { execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_006());}
}

/** Test Group 5: V2G timeouts */
if (PICS_CMN_CMN_V2gTimeouts) {

// SupportedAppProtocol ++++++
if (true) { execute(TC_EVCC_CMN_VTB_SupportedAppProtocol_003());}

// SessionSetup ++++++
if (true) { execute(TC_EVCC_CMN_VTB_SessionSetup_004());}

// ServiceDiscovery ++++++
if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_005());}

// ServiceDetail ++++++
if (PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_ServiceDetail_011());}

if (PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_ServiceDetail_012());}

// PaymentServiceSelection ++++++
if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009());}

// CertificateInstallation ++++++
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_009());}

// CertificateUpdate ++++++
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_011());}

// PaymentDetails ++++++
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005());}

// Authorization ++++++
if (true) { execute(TC_EVCC_CMN_VTB_Authorization_008());}

if (true) { execute(TC_EVCC_CMN_VTB_Authorization_009());}

// ChargeParameterDiscovery ++++++
if (true) { execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_007());}

if (true) { execute(TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001());}

// CableCheck ++++++
if (true) { execute(TC_EVCC_DC_VTB_CableCheck_005());}

if (true) { execute(TC_EVCC_DC_VTB_CableCheck_006());}

// PreCharge ++++++
if (true) { execute(TC_EVCC_DC_VTB_PreCharge_005());}

if (true) { execute(TC_EVCC_DC_VTB_PreCharge_006());}
}

```

```

// PowerDelivery Start ++++++
if (true) { execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009());}

// CurrentDemand ++++++
if (true) { execute(TC_EVCC_DC_VTB_CurrentDemand_005());}

// MeteringReceipt ++++++
if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_DC_VTB_MeteringReceipt_007());}

// PowerDelivery Stop ++++++
if (true) { execute(TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007());}

// WeldingDetection ++++++
if (PIXIT_EVCC_DC_WeldingDetection == weldingDetection) {
    execute(TC_EVCC_DC_VTB_WeldingDetection_005());}

// SessionStop ++++++
if (true) { execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005());}
}

/** Test Group 6: Invalid states and duty cycles */

if (PICS_EVCC_CMN_InvalidStatesAndDutyCycles) {

    // SessionSetup ++++++
    if (true) { execute(TC_EVCC_CMN_VTB_SessionSetup_006());}

    if (true) { execute(TC_EVCC_CMN_VTB_SessionSetup_007());}

    // ServiceDiscovery ++++++
    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_006());}

    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_007());}

    // ServiceDetail ++++++
    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == tls) {
        execute(TC_EVCC_CMN_VTB_ServiceDetail_015());}

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_ServiceDetail_016());}

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == eIM and
        PIXIT_EVCC_CMN_TLS == tls) {
        execute(TC_EVCC_CMN_VTB_ServiceDetail_017());}

    if (PIXIT_EVCC_CMN_VAS == serviceDetail and
        PICS_CMN_CMN_IdentificationMode == pnC and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_ServiceDetail_018());}

    // PaymentServiceSelection ++++++
    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_011());}

    if (true) { execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_012());}

    // CertificateInstallation ++++++
    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_CertificateHandling == certInstall and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {
        execute(TC_EVCC_CMN_VTB_CertificateInstallation_011());}

    if (PICS_CMN_CMN_IdentificationMode == pnC and
        PIXIT_EVCC_CMN_CertificateHandling == certInstall and
        PICS_CMN_CMN_PkiName == PKI_1 and
        PIXIT_CMN_CMN_PkiName == PKI_1) {

```

```

        execute(TC_EVCC_CMN_VTB_CertificateInstallation_012());

// CertificateUpdate ++++++
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_013());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_014());
}

// PaymentDetails ++++++
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_007());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_008());
}

// Authorization ++++++
if (true) { execute(TC_EVCC_CMN_VTB_Authorization_013());
}

if (true) { execute(TC_EVCC_CMN_VTB_Authorization_014());
}

// ChargeParameterDiscovery ++++++
if (true) { execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_009());
}

if (true) { execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_010());
}

// CableCheck ++++++
if (true) { execute(TC_EVCC_DC_VTB_CableCheck_008());
}

if (true) { execute(TC_EVCC_DC_VTB_CableCheck_009());
}

// PreCharge ++++++
if (true) { execute(TC_EVCC_DC_VTB_PreCharge_008());
}

if (true) { execute(TC_EVCC_DC_VTB_PreCharge_009());
}

// PowerDelivery Start ++++++
if (true) { execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_011());
}

if (true) { execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_011());
}

// CurrentDemand ++++++
if (true) { execute(TC_EVCC_DC_VTB_CurrentDemand_006());
}

if (true) { execute(TC_EVCC_DC_VTB_CurrentDemand_007());
}

// MeteringReceipt ++++++
if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_DC_VTB_MeteringReceipt_011());
}

if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_DC_VTB_MeteringReceipt_012());
}

// PowerDelivery Stop ++++++
if (true) { execute(TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_011());
}

if (true) { execute(TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_012());
}

// WeldingDetection ++++++
if (PIXIT_EVCC_DC_WeldingDetection == weldingDetection) {
    execute(TC_EVCC_DC_VTB_WeldingDetection_007());
}

if (PIXIT_EVCC_DC_WeldingDetection == weldingDetection) {

```

```

        execute(TC_EVCC_DC_VTB_WeldingDetection_008()));
    }
}

module DC_EVCC_Charging_Control {

    import from PICS_15118_2 all;
    import from PIXIT_15118_2 all;
    import from PICS_15118 all;
    import from PIXIT_15118 all;
    import from TestCases_EVCC_V2GTPSDP all;
    import from TestCases_EVCC_SDP all;
    import from TestCases_EVCC_SupportedAppProtocol all;
    import from TestCases_EVCC_V2GTPSessionSetup all;
    import from TestCases_EVCC_SessionSetup all;
    import from TestCases_EVCC_ServiceDiscovery all;
    import from TestCases_EVCC_ServiceDetailAndPaymentSelection all;
    import from TestCases_EVCC_ServiceDetail all;
    import from TestCases_EVCC_CertificateHandlingAndPaymentDetails all;
    import from TestCases_EVCC_CertificateInstallation all;
    import from TestCases_EVCC_CertificateUpdate all;
    import from TestCases_EVCC_PaymentDetails all;
    import from TestCases_EVCC_PaymentServiceSelection all;
    import from TestCases_EVCC_Authorization all;
    import from TestCases_EVCC_ChargeParameterDiscovery all;
    import from TestCases_EVCC_PowerDelivery all;
    import from TestCases_EVCC_MeteringReceipt all;
    import from TestCases_EVCC_CableCheck all;
    import from TestCases_EVCC_PreCharge all;
    import from TestCases_EVCC_PreChargeOrPowerDelivery all;
    import from TestCases_EVCC_CurrentDemand all;
    import from TestCases_EVCC_ChargingLoopDcOrPowerDelivery all;
    import from TestCases_EVCC_WeldingDetection all;
    import from TestCases_EVCC_WeldingDetectionOrSessionStop all;
    import from TestCases_EVCC_SessionStop all;
    import from ComponentsAndPorts all;
    import from Timer_15118 all;
    import from Timer_15118_2 all;

    control {

        // V2GTP explicit and SDP ++++++
        if (PICS_CMN_CMN_V2gtpSdp) { execute(TC_EVCC_CMN_VTB_V2GTPSDP_001()); }

        // SDP ++++++
        if (PICS_CMN_CMN_IdentificationMode == eIM and
            PIXIT_EVCC_CMN_TLS == none_) { execute(TC_EVCC_CMN_VTB_SDP_001()); }

        if (PICS_CMN_CMN_IdentificationMode == eIM and
            PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_SDP_002()); }

        if (PICS_CMN_CMN_IdentificationMode == eIM and
            PIXIT_EVCC_CMN_TLS == unknown) { execute(TC_EVCC_CMN_VTB_SDP_003()); }

        if (PICS_CMN_CMN_IdentificationMode == pnC) { execute(TC_EVCC_CMN_VTB_SDP_004()); }

        if (PICS_CMN_CMN_IdentificationMode == eIM and
            PIXIT_EVCC_CMN_TLS == none_ and
            PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
            PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
            execute(TC_EVCC_DC_VTB_SDP_001());
        }

        if (PICS_CMN_CMN_IdentificationMode == eIM and
            PIXIT_EVCC_CMN_TLS == tls and
            PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
            PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
            execute(TC_EVCC_DC_VTB_SDP_002());
        }

        if (PICS_CMN_CMN_IdentificationMode == eIM and
            PIXIT_EVCC_CMN_TLS == unknown and
            PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
            PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
            execute(TC_EVCC_DC_VTB_SDP_003());
        }
    }
}

```



```

if (PICS_CMN_CMN_IdentificationMode == pnC and
PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_DC_VTB_SDP_004());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_EVCC_CMN_TLS == none_ and
PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_DC_VTB_SDP_005());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_EVCC_CMN_TLS == tls and
PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_DC_VTB_SDP_006());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_EVCC_CMN_TLS == unknown and
PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_DC_VTB_SDP_007());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_DC_VTB_SDP_008());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_EVCC_CMN_TLS == none_ and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
PICS_CMN_CMN_CombinedTesting and
par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_DC_VTB_SDP_009());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_EVCC_CMN_TLS == tls and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
PICS_CMN_CMN_CombinedTesting and
par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_DC_VTB_SDP_010());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_EVCC_CMN_TLS == unknown and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
PICS_CMN_CMN_CombinedTesting and
par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_DC_VTB_SDP_011());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
PICS_CMN_CMN_CombinedTesting and
par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_DC_VTB_SDP_012());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_EVCC_CMN_TLS == none_ and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
PICS_CMN_CMN_CombinedTesting and
par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_DC_VTB_SDP_013());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_EVCC_CMN_TLS == tls and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
PICS_CMN_CMN_CombinedTesting and
par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_DC_VTB_SDP_014());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_EVCC_CMN_TLS == unknown and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
PICS_CMN_CMN_CombinedTesting and
par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
execute(TC_EVCC_DC_VTB_SDP_015());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
PICS_CMN_CMN_CombinedTesting and

```

```

    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
        execute(TC_EVCC_DC_VTB_SDP_016());};

if (true) { execute(TC_EVCC_CMN_VTB_SDP_006());};

// SupportedAppProtocol ++++++//
if ((PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == tls) or
    PICS_CMN_CMN_IdentificationMode == pnC) {
    execute(TC_EVCC_CMN_VTB_SupportedAppProtocol_001());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == none_) { execute(TC_EVCC_CMN_VTB_SupportedAppProtocol_004());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_TLS == unknown) { execute(TC_EVCC_CMN_VTB_SupportedAppProtocol_005());};

// V2GTPSessionSetup ++++++//
if (PICS_CMN_CMN_V2gtpV2g) { execute(TC_EVCC_CMN_VTB_V2GTPSessionSetup_001());};

// SessionSetup ++++++//
if (not PICS_CMN_CMN_CombinedTesting) { execute(TC_EVCC_CMN_VTB_SessionSetup_001());};

if (PICS_CMN_CMN_CombinedTesting) { execute(TC_EVCC_CMN_VTB_SessionSetup_008());};

if (PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_SessionSetup_001());};

if (PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_SessionSetup_002());};

if (PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_SessionSetup_003());};

if (PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_SessionSetup_004());};

// ServiceDiscovery ++++++//
if (true) { execute(TC_EVCC_CMN_VTB_ServiceDiscovery_001());};

// ServiceDetail ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_ServiceDetail_001());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PIXIT_EVCC_CMN_TLS == tls) { execute(TC_EVCC_CMN_VTB_ServiceDetail_002());};

// PaymentServiceSelection ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_PaymentServiceSelection_001());};

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PIXIT_EVCC_CMN_TLS == tls) {
    execute(TC_EVCC_CMN_VTB_PaymentServiceSelection_002());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == none_ and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {

```

```

        execute(TC_EVCC_CMN_VTB_PaymentServiceSelection_003());

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == none_) {
    execute(TC_EVCC_CMN_VTB_PaymentServiceSelection_004());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == unknown and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001());

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == unknown) {
    execute(TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_PaymentServiceSelection_001());

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PIXIT_EVCC_CMN_TLS == tls and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_PaymentServiceSelection_002());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == none_ and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_PaymentServiceSelection_003());

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == none_ and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_PaymentServiceSelection_004());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == unknown and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001());

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == unknown and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_PaymentServiceSelection_005());

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PIXIT_EVCC_CMN_TLS == tls and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_PaymentServiceSelection_006());

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == none_ and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and

```

```

PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_PaymentServiceSelection_007());
}

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == none_ and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_PaymentServiceSelection_008());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == unknown and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003());
}

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == unknown and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_PaymentServiceSelection_009());
}

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PIXIT_EVCC_CMN_TLS == tls and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_PaymentServiceSelection_010());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == none_ and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_PaymentServiceSelection_011());
}

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == none_ and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_PaymentServiceSelection_012());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == unknown and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005());
}

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == unknown and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006());
}

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {

```

```

        execute(TC_EVCC_DC_VTB_PaymentServiceSelection_013()));

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == serviceDetail and
    PIXIT_EVCC_CMN_TLS == tls and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_PaymentServiceSelection_014()));

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == none_ and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_PaymentServiceSelection_015()));

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == none_ and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_PaymentServiceSelection_016()));

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_VAS == unknown and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007()));

if (PICS_CMN_CMN_IdentificationMode == eIM and
    PIXIT_EVCC_CMN_VAS == unknown and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008()));

// CertificateInstallation ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateInstallation_001()));

// CertificateUpdate ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateUpdate_001()));

// PaymentDetails ++++++//
if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_PaymentDetails_001()));

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_PaymentDetails_002()));

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == none_ and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_PaymentDetails_003()));

if (PICS_CMN_CMN_IdentificationMode == pnC and

```

```

PIXIT_EVCC_CMN_CertificateHandling == unknown and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001());};

// Authorization ++++++//

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_Authorization_001());};

if (PICS_CMN_CMN_IdentificationMode == eIM) {
    execute(TC_EVCC_CMN_VTB_Authorization_002());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_Authorization_010());};

if (PICS_CMN_CMN_IdentificationMode == pnC and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_CMN_VTB_Authorization_011());};

// Start Charging Loop ++++++//

// ChargeParameterDiscovery ++++++//
if (PIXIT_EVCC_CMN_DepartureTime == unknown) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_001());};

if (PIXIT_EVCC_CMN_DepartureTime == none) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_012());};

if (PIXIT_EVCC_CMN_DepartureTime == unknown and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_015());};

if (PIXIT_EVCC_CMN_DepartureTime == none and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_016());};

if (PIXIT_EVCC_CMN_DepartureTime == departureTime and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_017());};

if (PIXIT_EVCC_CMN_DepartureTime == unknown and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_018());};

if (PIXIT_EVCC_CMN_DepartureTime == none and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_019());};

if (PIXIT_EVCC_CMN_DepartureTime == departureTime and
    PIXIT_EVCC_CMN_Pause == pause and PICS_CMN_CMN_CombinedTesting and
    PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_020());};

if (PIXIT_EVCC_CMN_DepartureTime == unknown and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_021());};

if (PIXIT_EVCC_CMN_DepartureTime == none and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_022());};

```



```

if (PIXIT_EVCC_CMN_DepartureTime == departureTime and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_023());};

if (PIXIT_EVCC_CMN_DepartureTime == unknown and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_024());};

if (PIXIT_EVCC_CMN_DepartureTime == none_ and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_025());};

if (PIXIT_EVCC_CMN_DepartureTime == departureTime and
    PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
    PICS_CMN_CMN_CombinedTesting and
    par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_026());};

// CableCheck ++++++//
if (true) { execute(TC_EVCC_DC_VTB_CableCheck_001());};

// PreCharge ++++++//
if (true) { execute(TC_EVCC_DC_VTB_PreCharge_001());};

// PreCharge or PowerDelivery ++++++//
if (PIXIT_EVCC_CMN_ChargingProfileOptimization == unknown) {
    execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001());};

if (PIXIT_EVCC_CMN_ChargingProfileOptimization == optimization and
    PICS_CMN_EVCC_SalesTariff) {
    execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015());};

if (PIXIT_EVCC_CMN_ChargingProfileOptimization == none_ and
    PICS_CMN_EVCC_SalesTariff) {
    execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016());};

// CurrentDemand ++++++//
if (true) { execute(TC_EVCC_DC_VTB_CurrentDemand_001());};

// MeteringReceipt ++++++//
if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_MeterReading == unknown and
    PIXIT_EVCC_CMN_SAScheduleTupleIDMR == unknown) {
    execute(TC_EVCC_DC_VTB_MeteringReceipt_001());};

if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
    PIXIT_CMN_CMN_MeterReading == unknown and
    PIXIT_EVCC_CMN_SAScheduleTupleIDMR == unknown) {
    execute(TC_EVCC_DC_VTB_MeteringReceipt_008());};

if (PICS_EVCC_CMN_ReceiptRequired and
    PICS_CMN_CMN_IdentificationMode == pnC and
    PICS_CMN_CMN_PkiName == PKI_1 and
    PIXIT_CMN_CMN_PkiName == PKI_1 and
    PIXIT_EVCC_CMN_CertificateHandling == certInstall and
    PIXIT_CMN_CMN_MeterReading == unknown and
    PIXIT_EVCC_CMN_SAScheduleTupleIDMR == unknown) {
    execute(TC_EVCC_DC_VTB_MeteringReceipt_009());};

if (PICS_EVCC_CMN_ReceiptRequired and

```

```

PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_MeterReading == meterReading and
PIXIT_EVCC_CMN_SAScheduleTupleIDMR == unknown) {
execute(TC_EVCC_DC_VTB_MeteringReceipt_013());};

if (PICS_EVCC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
PIXIT_CMN_CMN_MeterReading == meterReading and
PIXIT_EVCC_CMN_SAScheduleTupleIDMR == unknown) {
execute(TC_EVCC_DC_VTB_MeteringReceipt_014());};

if (PICS_EVCC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PIXIT_EVCC_CMN_CertificateHandling == certInstall and
PIXIT_CMN_CMN_MeterReading == meterReading and
PIXIT_EVCC_CMN_SAScheduleTupleIDMR == unknown) {
execute(TC_EVCC_DC_VTB_MeteringReceipt_015());};

if (PICS_EVCC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_MeterReading == unknown and
PIXIT_EVCC_CMN_SAScheduleTupleIDMR == sAScheduleTupleID) {
execute(TC_EVCC_DC_VTB_MeteringReceipt_016());};

if (PICS_EVCC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
PIXIT_CMN_CMN_MeterReading == unknown and
PIXIT_EVCC_CMN_SAScheduleTupleIDMR == sAScheduleTupleID) {
execute(TC_EVCC_DC_VTB_MeteringReceipt_017());};

if (PICS_EVCC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PIXIT_EVCC_CMN_CertificateHandling == certInstall and
PIXIT_CMN_CMN_MeterReading == unknown and
PIXIT_EVCC_CMN_SAScheduleTupleIDMR == sAScheduleTupleID) {
execute(TC_EVCC_DC_VTB_MeteringReceipt_018());};

if (PICS_EVCC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_MeterReading == meterReading and
PIXIT_EVCC_CMN_SAScheduleTupleIDMR == sAScheduleTupleID) {
execute(TC_EVCC_DC_VTB_MeteringReceipt_019());};

if (PICS_EVCC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PIXIT_EVCC_CMN_CertificateHandling == certUpdate and
PIXIT_CMN_CMN_MeterReading == meterReading and
PIXIT_EVCC_CMN_SAScheduleTupleIDMR == sAScheduleTupleID) {
execute(TC_EVCC_DC_VTB_MeteringReceipt_020());};

if (PICS_EVCC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1 and
PIXIT_EVCC_CMN_CertificateHandling == certInstall and
PIXIT_CMN_CMN_MeterReading == meterReading and
PIXIT_EVCC_CMN_SAScheduleTupleIDMR == sAScheduleTupleID) {
execute(TC_EVCC_DC_VTB_MeteringReceipt_021());};

```

```
// PowerDelivery Stop ++++++
```

```

if (PICS_CMN_CMN_IdentificationMode == eIM or
(not PICS_EVCC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1)) {
    execute(TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001());};

if (PICS_EVCC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002());};

// PowerDelivery Renegotiate ++++++//
if (PICS_CMN_CMN_Renegotiation and
PIXIT_CMN_CMN_Renegotiation == unknown and
PICS_CMN_CMN_IdentificationMode == eIM or
(not PICS_EVCC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC)) {
    execute(TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008());};

if (PICS_CMN_CMN_Renegotiation and
PIXIT_CMN_CMN_Renegotiation == unknown and
PICS_EVCC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009());};

if (not PICS_CMN_CMN_Renegotiation and
PIXIT_CMN_CMN_Renegotiation == renegotiation and
PICS_CMN_CMN_IdentificationMode == eIM or
(not PICS_EVCC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC)) {
    execute(TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013());};

if (not PICS_CMN_CMN_Renegotiation and
PIXIT_CMN_CMN_Renegotiation == renegotiation and
PICS_EVCC_CMN_ReceiptRequired and
PICS_CMN_CMN_IdentificationMode == pnC and
PICS_CMN_CMN_PkiName == PKI_1 and
PIXIT_CMN_CMN_PkiName == PKI_1) {
    execute(TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014());};

// ChargeParameter after Renegotiation ++++++//
if (PICS_CMN_CMN_Renegotiation and
PIXIT_CMN_CMN_Renegotiation == unknown and
PIXIT_EVCC_CMN_DepartureTime == unknown) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_008());};

if (not PICS_CMN_CMN_Renegotiation and
PIXIT_CMN_CMN_Renegotiation == renegotiation and
PIXIT_EVCC_CMN_DepartureTime == unknown) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_011());};

if (PICS_CMN_CMN_Renegotiation and
PIXIT_CMN_CMN_Renegotiation == unknown and
PIXIT_EVCC_CMN_DepartureTime == none_) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_013());};

if (not PICS_CMN_CMN_Renegotiation and
PIXIT_CMN_CMN_Renegotiation == renegotiation and
PIXIT_EVCC_CMN_DepartureTime == none_) {
    execute(TC_EVCC_DC_VTB_ChargeParameterDiscovery_014());};

// PowerDelivery Start after Renegotiation ++++++//
if (PICS_CMN_CMN_Renegotiation and
PIXIT_CMN_CMN_Renegotiation == unknown and
PIXIT_EVCC_CMN_ChargingProfileOptimization == unknown and
PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
    execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013());};

if (not PICS_CMN_CMN_Renegotiation and
PIXIT_CMN_CMN_Renegotiation == renegotiation and
PIXIT_EVCC_CMN_ChargingProfileOptimization == unknown and
PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {

```

```

        execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014());

    if (PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == unknown and
        PIXIT_EVCC_CMN_ChargingProfileOptimization == optimization and
        PICS_CMN_EVCC_SalesTariff and
        PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
        execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017());
    }

    if (not PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == renegotiation and
        PIXIT_EVCC_CMN_ChargingProfileOptimization == optimization and
        PICS_CMN_EVCC_SalesTariff and
        PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
        execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018());
    }

    if (PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == unknown and
        PIXIT_EVCC_CMN_ChargingProfileOptimization == none_ and
        PICS_CMN_EVCC_SalesTariff and
        PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
        execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019());
    }

    if (not PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == renegotiation and
        PIXIT_EVCC_CMN_ChargingProfileOptimization == none_ and
        PICS_CMN_EVCC_SalesTariff and
        PIXIT_EVCC_CMN_NewSAScheduleTupleID == unknown) {
        execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020());
    }

    if (PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == unknown and
        PIXIT_EVCC_CMN_ChargingProfileOptimization == unknown and
        PIXIT_EVCC_CMN_NewSAScheduleTupleID == newSAScheduleTupleID) {
        execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021());
    }

    if (not PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == renegotiation and
        PIXIT_EVCC_CMN_ChargingProfileOptimization == unknown and
        PIXIT_EVCC_CMN_NewSAScheduleTupleID == newSAScheduleTupleID) {
        execute(TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022());
    }

    // End Charging Loop ++++++//

    // Welding Detection ++++++//
    if (PIXIT_EVCC_DC_WeldingDetection == weldingDetection) {
        execute(TC_EVCC_DC_VTB_WeldingDetection_001());
    }

    if (PIXIT_EVCC_DC_WeldingDetection == weldingDetection) {
        execute(TC_EVCC_DC_VTB_WeldingDetection_009());
    }

    // SessionStop ++++++//
    if (((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls) or
        PICS_CMN_CMN_IdentificationMode == pnC) and
        PIXIT_EVCC_DC_WeldingDetection == weldingDetection and
        PIXIT_EVCC_CMN_Pause == unknown) {
        execute(TC_EVCC_DC_VTB_SessionStop_001());
    }

    if (((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls) or
        PICS_CMN_CMN_IdentificationMode == pnC) and
        PIXIT_EVCC_DC_WeldingDetection == none_ and
        PIXIT_EVCC_CMN_Pause == unknown) {
        execute(TC_EVCC_DC_VTB_SessionStop_002());
    }

    if (((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls) or
        PICS_CMN_CMN_IdentificationMode == pnC) and
        PIXIT_EVCC_DC_WeldingDetection == unknown and
        PIXIT_EVCC_CMN_Pause == unknown) {
        execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001());
    }

    if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_) and
        PIXIT_EVCC_DC_WeldingDetection == weldingDetection and
        PIXIT_EVCC_CMN_Pause == unknown) {
        execute(TC_EVCC_DC_VTB_SessionStop_003());
    }

    if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_) and
        PIXIT_EVCC_DC_WeldingDetection == none_ and
        PIXIT_EVCC_CMN_Pause == unknown) {

```

```

        execute(TC_EVCC_DC_VTB_SessionStop_004());

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_) and
    PIXIT_EVCC_DC_WeldingDetection == unknown and
    PIXIT_EVCC_CMN_Pause == unknown) {
    execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007());

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == unknown) and
    PIXIT_EVCC_DC_WeldingDetection == weldingDetection and
    PIXIT_EVCC_CMN_Pause == unknown) {
    execute(TC_EVCC_DC_VTB_SessionStop_005());

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == unknown) and
    PIXIT_EVCC_DC_WeldingDetection == none_ and
    PIXIT_EVCC_CMN_Pause == unknown) {
    execute(TC_EVCC_DC_VTB_SessionStop_006());

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == unknown) and
    PIXIT_EVCC_DC_WeldingDetection == unknown and
    PIXIT_EVCC_CMN_Pause == unknown) {
    execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008());

if (((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls) or
    PICS_CMN_CMN_IdentificationMode == pnC) and
    PIXIT_EVCC_DC_WeldingDetection == weldingDetection and
    PIXIT_EVCC_CMN_Pause == none_) {
    execute(TC_EVCC_DC_VTB_SessionStop_007());

if (((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls) or
    PICS_CMN_CMN_IdentificationMode == pnC) and
    PIXIT_EVCC_DC_WeldingDetection == none_ and
    PIXIT_EVCC_CMN_Pause == none_) {
    execute(TC_EVCC_DC_VTB_SessionStop_008());

if (((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls) or
    PICS_CMN_CMN_IdentificationMode == pnC) and
    PIXIT_EVCC_DC_WeldingDetection == unknown and
    PIXIT_EVCC_CMN_Pause == none_) {
    execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009());

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_) and
    PIXIT_EVCC_DC_WeldingDetection == weldingDetection and
    PIXIT_EVCC_CMN_Pause == none_) {
    execute(TC_EVCC_DC_VTB_SessionStop_009());

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_) and
    PIXIT_EVCC_DC_WeldingDetection == none_ and
    PIXIT_EVCC_CMN_Pause == none_) {
    execute(TC_EVCC_DC_VTB_SessionStop_010());

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_) and
    PIXIT_EVCC_DC_WeldingDetection == unknown and
    PIXIT_EVCC_CMN_Pause == none_) {
    execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010());

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == unknown) and
    PIXIT_EVCC_DC_WeldingDetection == weldingDetection and
    PIXIT_EVCC_CMN_Pause == none_) {
    execute(TC_EVCC_DC_VTB_SessionStop_011());

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == unknown) and
    PIXIT_EVCC_DC_WeldingDetection == none_ and
    PIXIT_EVCC_CMN_Pause == none_) {
    execute(TC_EVCC_DC_VTB_SessionStop_012());

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == unknown) and
    PIXIT_EVCC_DC_WeldingDetection == unknown and
    PIXIT_EVCC_CMN_Pause == none_) {
    execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011());

if (((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls) or
    PICS_CMN_CMN_IdentificationMode == pnC) and
    PIXIT_EVCC_DC_WeldingDetection == weldingDetection and
    PIXIT_EVCC_CMN_Pause == pause) {
    execute(TC_EVCC_DC_VTB_SessionStop_013());

if (((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls) or
    PICS_CMN_CMN_IdentificationMode == pnC) and
    PIXIT_EVCC_DC_WeldingDetection == none_ and

```

```

    PIXIT_EVCC_CMN_Pause == pause) {
        execute(TC_EVCC_DC_VTB_SessionStop_014());
    }

    if (((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls) or
        PICS_CMN_CMN_IdentificationMode == pnC) and
        PIXIT_EVCC_DC_WeldingDetection == unknown and
        PIXIT_EVCC_CMN_Pause == pause) {
        execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012());
    }

    if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_) and
        PIXIT_EVCC_DC_WeldingDetection == weldingDetection and
        PIXIT_EVCC_CMN_Pause == pause) {
        execute(TC_EVCC_DC_VTB_SessionStop_015());
    }

    if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_) and
        PIXIT_EVCC_DC_WeldingDetection == none_ and
        PIXIT_EVCC_CMN_Pause == pause) {
        execute(TC_EVCC_DC_VTB_SessionStop_016());
    }

    if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_) and
        PIXIT_EVCC_DC_WeldingDetection == unknown and
        PIXIT_EVCC_CMN_Pause == pause) {
        execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013());
    }

    if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == unknown) and
        PIXIT_EVCC_DC_WeldingDetection == weldingDetection and
        PIXIT_EVCC_CMN_Pause == pause) {
        execute(TC_EVCC_DC_VTB_SessionStop_017());
    }

    if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == unknown) and
        PIXIT_EVCC_DC_WeldingDetection == none_ and
        PIXIT_EVCC_CMN_Pause == pause) {
        execute(TC_EVCC_DC_VTB_SessionStop_018());
    }

    if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == unknown) and
        PIXIT_EVCC_DC_WeldingDetection == unknown and
        PIXIT_EVCC_CMN_Pause == pause) {
        execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014());
    }

    if (((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls) or
        PICS_CMN_CMN_IdentificationMode == pnC) and
        PIXIT_EVCC_DC_WeldingDetection == weldingDetection and
        PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge) {
        execute(TC_EVCC_DC_VTB_SessionStop_019());
    }

    if (((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls) or
        PICS_CMN_CMN_IdentificationMode == pnC) and
        PIXIT_EVCC_DC_WeldingDetection == none_ and
        PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge) {
        execute(TC_EVCC_DC_VTB_SessionStop_020());
    }

    if (((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls) or
        PICS_CMN_CMN_IdentificationMode == pnC) and
        PIXIT_EVCC_DC_WeldingDetection == unknown and
        PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge) {
        execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015());
    }

    if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_) and
        PIXIT_EVCC_DC_WeldingDetection == weldingDetection and
        PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge) {
        execute(TC_EVCC_DC_VTB_SessionStop_021());
    }

    if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_) and
        PIXIT_EVCC_DC_WeldingDetection == none_ and
        PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge) {
        execute(TC_EVCC_DC_VTB_SessionStop_022());
    }

    if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_) and
        PIXIT_EVCC_DC_WeldingDetection == unknown and
        PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge) {
        execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016());
    }

    if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == unknown) and
        PIXIT_EVCC_DC_WeldingDetection == weldingDetection and
        PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge) {
        execute(TC_EVCC_DC_VTB_SessionStop_023());
    }

    if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == unknown) and
        PIXIT_EVCC_DC_WeldingDetection == none_ and

```



```

PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge) {
execute(TC_EVCC_DC_VTB_SessionStop_024());};

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == unknown) and
PIXIT_EVCC_DC_WeldingDetection == unknown and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge) {
execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017());};

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls) or
PICS_CMN_CMN_IdentificationMode == pnC) and
PIXIT_EVCC_DC_WeldingDetection == weldingDetection and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge) {
execute(TC_EVCC_DC_VTB_SessionStop_025());};

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls) or
PICS_CMN_CMN_IdentificationMode == pnC) and
PIXIT_EVCC_DC_WeldingDetection == none_ and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge) {
execute(TC_EVCC_DC_VTB_SessionStop_026());};

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls) or
PICS_CMN_CMN_IdentificationMode == pnC) and
PIXIT_EVCC_DC_WeldingDetection == unknown and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge) {
execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018());};

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_) and
PIXIT_EVCC_DC_WeldingDetection == weldingDetection and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge) {
execute(TC_EVCC_DC_VTB_SessionStop_027());};

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_) and
PIXIT_EVCC_DC_WeldingDetection == none_ and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge) {
execute(TC_EVCC_DC_VTB_SessionStop_028());};

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_) and
PIXIT_EVCC_DC_WeldingDetection == unknown and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge) {
execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019());};

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == unknown) and
PIXIT_EVCC_DC_WeldingDetection == weldingDetection and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge) {
execute(TC_EVCC_DC_VTB_SessionStop_029());};

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == unknown) and
PIXIT_EVCC_DC_WeldingDetection == none_ and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge) {
execute(TC_EVCC_DC_VTB_SessionStop_030());};

if ((PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == unknown) and
PIXIT_EVCC_DC_WeldingDetection == unknown and
PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge) {
execute(TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020());};
}
}

```

Annex C (normative)

Test-case specifications for 15118-2 V2GTP

This annex covers all V2GTP test case *specifications* in the ATS.

C.1 SECC test cases

```

module TestCases_SECC_V2GTPSessionSetup {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_V2GTPSessionSetup all;
    import from Services_HAL_61851 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from Templates_CMN_V2GTPMessage all;

    testcase TC_SECC_CMN_VTB_V2GTPSessionSetup_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict == pass) {
            f_SECC_CMN_TB_VTB_V2GTPSessionSetup_001();
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        tc_V2G_EVCC_CommunicationSetup_Timer.stop;
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_V2GTPSessionSetup_002() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict == pass) {
            f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
                '01'H,'FE'H,'8001'H, '00000000'H));
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        tc_V2G_EVCC_CommunicationSetup_Timer.stop;
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_V2GTPSessionSetup_003() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
    }
}

```

```

var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict == pass) {
    f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
        '01'H,'FE'H,'7001'H, omit));
}
else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
tc_V2G_EVCC_CommunicationSetup_Timer.stop;
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_V2GTPSessionSetup_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
            'FF'H,'FE'H,'8001'H, omit));
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_V2GTPSessionSetup_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
            '01'H,'FF'H,'8001'H, omit));
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

}

module TestCases_SECC_V2GTPSDP {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851 all;
    import from TestBehavior_SECC_V2GTPSDP all;

```

```

import from TestBehavior_SECC_CommonBehavior all;
import from Templates_CMN_V2GTPMessage all;
import from Preconditions_SECC_15118_2 all;
import from PostConditions_SECC_15118_2 all;
import from ComponentsAndPorts all;
import from Configurations_15118_2 all;

testcase TC_SECC_CMN_VTB_V2GTPSDP_001() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_V2GTPSDP_001();
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_V2GTPSDP_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
            '01'H,'FE'H,'9000'H, '00000000'H));
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_V2GTPSDP_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
            '01'H,'FE'H,'7000'H, omit));
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_V2GTPSDP_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

```

```

//----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict == pass) {
    f_SECC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
        'FF'H,'FE'H,'9000'H, omit));
}
else{
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_V2GTPSDP_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
            '01'H,'FF'H,'9000'H, omit));
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

C.2 EVCC test cases

```

module TestCases_EVCC_V2GTPSessionSetup {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_V2GTPSessionSetup all;
    import from Templates_CMN_V2GTPMessage all;

    testcase TC_EVCC_CMN_VTB_V2GTPSessionSetup_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_001();
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_V2GTPSessionSetup_002() runs on EVCC_Tester system SystemEVCC {

```

```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
        '01'H, 'FE'H, '8001'H, '00000000'H));
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_V2GTPSessionSetup_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
            '01'H, 'FE'H, '7001'H, omit));
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_V2GTPSessionSetup_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
            'FF'H, 'FE'H, '8001'H, omit));
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_V2GTPSessionSetup_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
            '01'H, 'FF'H, '8001'H, omit));
    }else {
        log("PreCondition was unsuccessful.");
    }
}

```



```

    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

}

module TestCases_EVCC_V2GTPSDP {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_V2GTPSDP all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Preconditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from Templates_CMN_V2GTPMessage all;

    testcase TC_EVCC_CMN_VTB_V2GTPSDP_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict == pass) {
            f_EVCC_CMN_TB_VTB_V2GTPSDP_001();
        }
        else{
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_V2GTPSDP_002() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict == pass) {
            f_EVCC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
                '01'H, 'FE'H, '9001'H, '00000000'H));
        }
        else{
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_V2GTPSDP_003() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        //----- Test behavior-----
    }
}

```

```

    if(preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
            '01'H, 'FE'H, '7001'H, omit));
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_V2GTPSDP_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
            'FF'H, 'FE'H, '9001'H, omit));
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_V2GTPSDP_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
            '01'H, 'FF'H, '9001'H, omit));
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

Annex D (normative)

Test-case specifications for 15118-2 SDP messages

This annex covers all SDP test case *specifications* in the ATS.

D.1 SECC test cases

```

module TestCases_SECC_SDP {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851 all;
    import from TestBehavior_SECC_SDP all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Preconditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_SECC_CMN_VTB_SDP_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict == pass) {
            f_SECC_CMN_TB_VTB_SDP_001('10'H, fail);
        }
        else{
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_SDP_002() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict == pass) {
            f_SECC_CMN_TB_VTB_SDP_001('00'H, fail);
        }
        else{
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_SDP_003() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    }
}

```

```

preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict == pass) {
    f_SECC_CMN_TB_VTB_SDP_001('00'H, fail);
}
else{
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_SDP_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SDP_002('10'H);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_SDP_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SDP_002('00'H);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_SDP_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SDP_002('00'H);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

}

testcase TC_SECC_CMN_VTB_SDP_007() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SDP_003();
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

D.2 EVCC test cases

```

module TestCases_EVCC_SDP {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_SDP all;
    import from PostConditions_EVCC_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;

    testcase TC_EVCC_CMN_VTB_SDP_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict == pass) {
            f_EVCC_CMN_TB_VTB_SDP_001('10'H, fail);
        }
        else{
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----

        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_SDP_002() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict == pass) {
            f_EVCC_CMN_TB_VTB_SDP_001('00'H, fail);

```

```

    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----

    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SDP_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_SDP_001(?, fail);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----

    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SDP_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_SDP_001('00'H, fail);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----

    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SDP_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_SDP_002();
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```



```

testcase TC_EVCC_CMN_VTB_SDP_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_SDP_003();
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SDP_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_SDP_004();
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_001() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);

```

```

    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_EVCC_AC_VTB_SDP_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;

```

```

var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
}

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_SDP_002();
    if(getverdict == pass) {
        setverdict(pass, "The SUT has executed a reset of the message counter " &
            "after previous SDP process.");
    }
} else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass){
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----

```

```

f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
}

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_SDP_002();
    if(getverdict == pass) {
        setverdict(pass, "The SUT has executed a reset of the message counter " &
            "after previous SDP process.");
    }
}
else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

```

```

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
}

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_SDP_002();
    if(getverdict == pass) {
        setverdict(pass, "The SUT has executed a reset of the message counter " &
            "after previous SDP process.");
    }
} else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_012() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }
}

```



```

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
}

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_SDP_002();
    if(getverdict == pass) {
        setverdict(pass, "The SUT has executed a reset of the message counter " &
            "after previous SDP process.");
    }
} else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_013() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_014() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);

```

```

        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_015() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_016() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }
}

```

```

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_001() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {

```

```

    preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_SDP_002();
    if(getverdict == pass) {
        setverdict(pass, "The SUT has executed a reset of the message counter " &
            "after previous SDP process.");
    }
}
}
else {
    log("PreCondition was unsuccessful.");
}
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }
    }
    else {
        log("PreCondition was unsuccessful.");
    }
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }
}

```

```

    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_005() runs on EVCC_Tester system SystemEVCC {
    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_006() runs on EVCC_Tester system SystemEVCC {
    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }
}

```

```

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_SDP_002();
    if(getverdict == pass) {
        setverdict(pass, "The SUT has executed a reset of the message counter " &
            "after previous SDP process.");
    }
}
else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass){
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }
}

```



```

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_SDP_002();
    if(getverdict == pass) {
        setverdict(pass, "The SUT has executed a reset of the message counter " &
            "after previous SDP process.");
    }
}
else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----

```

```

    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_012() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){

```

```

        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                           "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_013() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                           "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_014() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();

```

```

        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_015() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdictctype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_016() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdictctype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {

```

```

        setverdict(pass, "The SUT has executed a reset of the message counter " &
                      "after previous SDF process.");
    }
} else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

STANDARDSISO.COM : Click to view the full PDF of ISO 15118-4:2018

Annex E (normative)

Test-case specifications for 15118-2 V2G application layer messages

This annex covers all V2G application layer test case *specifications* in the ATS.

E.1 SECC test cases

E.1.1 V2G protocol handshake

E.1.1.1 SECC test cases for SupportedAppProtocol

```

module TestCases_SECC_SupportedAppProtocol {

    import from Services_HAL_61851 all;
    import from TestBehavior_SECC_SupportedAppProtocol all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from urn_iso_15118_2_2010_AppProtocol all;
    import from Timer_15118_2 all;
    import from TestBehavior_SECC_CommonBehavior all;

    testcase TC_SECC_CMN_VTB_SupportedAppProtocol_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_Binding_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_SECC_CMN_PR_InitConfiguration_002(system, pass, fail);
            f_SECC_CMN_TB_VTB_SupportedAppProtocol_001(fail);
            tc_V2G_EVCC_CommunicationSetup_Timer.stop;
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_SupportedAppProtocol_002() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_Binding_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_SECC_CMN_PR_InitConfiguration_002(system, pass, fail);
            f_SECC_CMN_TB_VTB_SupportedAppProtocol_002(fail);
            tc_V2G_EVCC_CommunicationSetup_Timer.stop;
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    }
}

```



```

    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_SupportedAppProtocol_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Binding_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_PR_InitConfiguration_002(system, pass, fail);
        f_SECC_CMN_TB_VTB_SupportedAppProtocol_003();
        tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_SupportedAppProtocol_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Binding_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_PR_InitConfiguration_002(system, pass, fail);
        f_SECC_CMN_TB_VTB_SupportedAppProtocol_001(fail);
        tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_SupportedAppProtocol_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Binding_001(v_HAL_61851_Listener);

    alt {
        [] tc_V2G_SECC_CommunicationSetup_Timer.timeout;
        [] pt_V2G_UDP_SDP_Port.receive {
            setverdict(fail, "No SDP message was expected.");
        }
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_PR_InitConfiguration_002(system, fail, pass);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_SECC_CMN_VTB_SupportedAppProtocol_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Binding_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_PR_InitConfiguration_002(system, pass, fail);
        f_SECC_CMN_TB_VTB_SupportedAppProtocol_004();
        tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.1.2 V2G messages

E.1.2.1 SECC test cases for SessionSetup

```

module TestCases_SECC_SessionSetup {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_SessionSetup all;
    import from Services_HAL_61851 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from LibFunctions_15118_2 all;
    import from Timer_15118_2 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;

    testcase TC_SECC_CMN_VTB_SessionSetup_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict == pass) {
            f_SECC_CMN_TB_VTB_SessionSetup_001(fail);
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        tc_V2G_EVCC_CommunicationSetup_Timer.stop;
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_SessionSetup_002() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
    }
}

```

```

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict == pass) {
    f_SECC_CMN_TB_VTB_SessionSetup_002();
}
else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
tc_V2G_EVCC_CommunicationSetup_Timer.stop;
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_SessionSetup_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_003();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_SessionSetup_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_004();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_SessionSetup_001() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }
}

```

```

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
    preConVerdict :=
        f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict == pass) {
    f_SECC_CMN_TB_VTB_SessionSetup_005(fail);
}
else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
tc_V2G_EVCC_CommunicationSetup_Timer.stop;
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_SessionSetup_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict :=
            f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_006(false);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_SessionSetup_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict :=
            f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_006(true);
    }
}

```

```

else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
tc_V2G_EVCC_CommunicationSetup_Timer.stop;
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_SessionSetup_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict :=
            f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_005(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_SessionSetup_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict :=
            f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_006(false);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_SessionSetup_006() runs on SECC_Tester system SystemSECC {

```

```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
    preConVerdict :=
        f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict == pass) {
    f_SECC_CMN_TB_VTB_SessionSetup_006(true);
}
else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
tc_V2G_EVCC_CommunicationSetup_Timer.stop;
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionSetup_001() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict :=
            f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_005(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionSetup_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

```



```

    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict :=
            f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_006(false);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionSetup_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict :=
            f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_006(true);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionSetup_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict :=
            f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_005(fail);
    }
}

```

```

    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionSetup_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict :=
            f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_006(false);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionSetup_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict :=
            f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_006(true);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

}

E.1.2.2 SECC test cases for ServiceDiscovery

```

module TestCases_SECC_ServiceDiscovery {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_ServiceDiscovery all;
    import from Services_HAL_61851 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Preconditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from LibFunctions_15118_2 all;
    import from Timer_15118_2 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;

    testcase TC_SECC_CMN_VTB_ServiceDiscovery_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_ServiceDiscovery_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_ServiceDiscovery_002() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_ServiceDiscovery_002();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_ServiceDiscovery_003() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_ServiceDiscovery_003();
        }else {
            log("PreCondition was unsuccessful.");
        }
    }
}

```

```

    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ServiceDiscovery_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDiscovery_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ServiceDiscovery_001() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDiscovery_005(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ServiceDiscovery_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----

```

```

    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDiscovery_005(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ServiceDiscovery_001() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDiscovery_005(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ServiceDiscovery_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDiscovery_005(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.1.2.3 SECC test cases for ServiceDetail

```
module TestCases_SECC_ServiceDetail {
```

```

import from DataStructure_HAL_61851 all;
import from TestBehavior_SECC_ServiceDetail all;
import from Services_HAL_61851 all;
import from TestBehavior_SECC_CommonBehavior all;
import from PreConditions_SECC_15118_2 all;
import from PostConditions_SECC_15118_2 all;
import from ComponentsAndPorts all;
import from Configurations_15118_2 all;

testcase TC_SECC_CMN_VTB_ServiceDetail_001() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDetail_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ServiceDetail_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDetail_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ServiceDetail_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDetail_002();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ServiceDetail_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----

```



```

f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_CMN_TB_VTB_ServiceDetail_002();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ServiceDetail_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDetail_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ServiceDetail_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDetail_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ServiceDetail_007() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDetail_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ServiceDetail_008() runs on SECC_Tester system SystemSECC {

```

```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_CMN_TB_VTB_ServiceDetail_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ServiceDetail_009() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDetail_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ServiceDetail_010() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDetail_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ServiceDetail_011() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDetail_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----

```

```

    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ServiceDetail_012() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDetail_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.1.2.4 SECC test cases for PaymentServiceSelection

```

module TestCases_SECC_PaymentServiceSelection {

    import from Pics_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_PaymentServiceSelection all;
    import from Services_HAL_61851 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_SECC_CMN_VTB_PaymentServiceSelection_001() runs on SECC_Tester
                                                         system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001 (v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if ( preConVerdict == pass ) {
            f_SECC_CMN_TB_VTB_PaymentServiceSelection_001(fail);
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_PaymentServiceSelection_002() runs on SECC_Tester
                                                         system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001 (v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if ( preConVerdict == pass ) {
            f_SECC_CMN_TB_VTB_PaymentServiceSelection_001(fail);
        }
    }
}

```

```

else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentServiceSelection_003() runs on SECC_Tester
                                     system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001 (v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_CMN_TB_VTB_PaymentServiceSelection_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentServiceSelection_004() runs on SECC_Tester
                                     system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001 (v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_CMN_TB_VTB_PaymentServiceSelection_002();
    }
    else {
        log ("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001 ( v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentServiceSelection_005() runs on SECC_Tester
                                     system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001 ( v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_CMN_TB_VTB_PaymentServiceSelection_003();
    }
    else {
        log ("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001 ( v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentServiceSelection_006() runs on SECC_Tester

```

```

system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001( v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

//----- Test behavior-----
if ( preConVerdict == pass ) {
    f_SECC_CMN_TB_VTB_PaymentServiceSelection_004();
}
else {
    log ("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001( v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentServiceSelection_007() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001( v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

//----- Test behavior-----
if ( preConVerdict == pass ) {
    f_SECC_CMN_TB_VTB_PaymentServiceSelection_005();
}
else {
    log ("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001( v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentServiceSelection_008() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001( v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

//----- Test behavior-----
if ( preConVerdict == pass ) {
    f_SECC_CMN_TB_VTB_PaymentServiceSelection_006();
}
else {
    log ("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001( v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentServiceSelection_009() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001( v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

//----- Test behavior-----

```

```

if ( preConVerdict == pass ) {
    f_SECC_CMN_TB_VTB_PaymentServiceSelection_007();
}
else {
    log ("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001( v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentServiceSelection_010() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001( v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_CMN_TB_VTB_PaymentServiceSelection_007();
    }
    else {
        log ("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001( v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.1.2.5 SECC test cases for PaymentDetails

```

module TestCases_SECC_PaymentDetails {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_PaymentDetails all;
    import from Services_HAL_61851 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from Templates_CMN_Security all;

    testcase TC_SECC_CMN_VTB_PaymentDetails_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_PaymentDetails_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_PaymentDetails_002() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----

```



```

f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_CMN_TB_VTB_PaymentDetails_001(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentDetails_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_PaymentDetails_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentDetails_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_PaymentDetails_002();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentDetails_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_PaymentDetails_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentDetails_006() runs on SECC_Tester system SystemSECC {

```

```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_CMN_TB_VTB_PaymentDetails_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentDetails_007() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_PaymentDetails_005("CRT CONTRACT LEAF EXPIRED",
                                              "CRT_MO_SUB2_VALID",
                                              "CRT_MO_SUB1_VALID",
                                              FAILED_CertificateExpired);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentDetails_008() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_PaymentDetails_005("CRT CONTRACT LEAF INVALID_CONTENT",
                                              "CRT_MO_SUB2_VALID",
                                              "CRT_MO_SUB1_VALID",
                                              m_CMN_failedResponseCodeList_003());
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentDetails_009() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){

```

```

        f_SECC_CMN_TB_VTB_PaymentDetails_005(
            "CRT_CONTRACT_LEAF_EXPIRED_ISSUER_SUB1",
            "CRT_MO_SUB2_EXPIRED",
            "CRT_MO_SUB1_VALID",
            m_CMN_failedResponseCodeList_003());
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentDetails_010() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_PaymentDetails_005(
            "CRT_CONTRACT_LEAF_EXPIRED_ISSUER_SUB2",
            "CRT_MO_SUB2_VALID",
            "CRT_MO_SUB1_EXPIRED",
            m_CMN_failedResponseCodeList_003());
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.1.2.6 SECC test cases for Authorization

```

module TestCases_SECC_Authorization {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_Authorization all;
    import from Services_HAL_61851 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Preconditions_SECC_15118_2 all;
    import from Preconditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_SECC_CMN_VTB_Authorization_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_Authorization_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_Authorization_002() runs on SECC_Tester system SystemSECC {

```

```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_CMN_TB_VTB_Authorization_002(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----

```

```

    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_007() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_006();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_008() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_007();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_009() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_007();

```

```

}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_010() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_008();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_011() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_009();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_012() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_002(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_013() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);

```

```

preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_CMN_TB_VTB_Authorization_002(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_014() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_010();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_015() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_011();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_016() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_011();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_017() runs on SECC_Tester system SystemSECC {

```



```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_CMN_TB_VTB_Authorization_011();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.1.2.7 SECC test cases for ChargeParameterDiscovery

```

module TestCases_SECC_ChargeParameterDiscovery {

import from DataStructure_HAL_61851 all;
import from TestBehavior_SECC_ChargeParameterDiscovery all;
import from ComponentsAndPorts all;
import from PreConditions_SECC_15118_2 all;
import from PostConditions_SECC_15118_2 all;
import from Configurations_15118_2 all;
import from LibFunctions_15118_2 all;
import from Timer_15118_2 all;
import from TestBehavior_SECC_CommonBehavior all;
import from Timer_15118 all;
import from Pics_15118 all;
import from Pixit_15118 all;

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_001() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_002() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
}else {
    log("PreCondition was unsuccessful.");
}
}
}

```

```

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_003() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_002();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_004() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_003();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_005() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_006() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);

```

```

preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_007() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_008() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_009() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----

```

```

    if(preConVerdict==pass){
        vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_010() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_011() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(FICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
    }else {
        log("PreCondition was unsuccessful.");
    }
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_012() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
    }
}

```

```

    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_013() runs on SECC_Tester
                                                    system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_014() runs on SECC_Tester
                                                    system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_008();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_015() runs on SECC_Tester
                                                    system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_016() runs on SECC_Tester
                                                    system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;

```

```

var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_017() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
    preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_018() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
    preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);

```

```

    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_019() runs on SECC_Tester
                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_020() runs on SECC_Tester
                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_021() runs on SECC_Tester
                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);

```



```

    f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
    preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_022() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_023() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_024() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;

```

```

var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_025() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_026() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_027() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
}

```

```

    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_028() runs on SECC_Tester
                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_029() runs on SECC_Tester
                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_030() runs on SECC_Tester
                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }
}

```

```

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
    preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_031() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_032() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_033() runs on SECC_Tester
                                                    system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_034() runs on SECC_Tester
                                                    system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_035() runs on SECC_Tester
                                                    system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

```

```

    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_036() runs on SECC_Tester
                                                    system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp, 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_001() runs on SECC_Tester
                                                    system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_002() runs on SECC_Tester
                                                    system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

```

```

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_DC_TB_VTB_ChargeParameterDiscovery_002();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_003() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_004() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_005() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----

```



```

    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_006() runs on SECC_Tester
    system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_007() runs on SECC_Tester
    system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_008() runs on SECC_Tester
    system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_009() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_010() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_011() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_012() runs on SECC_Tester

```

```

system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_013() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_014() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_DC_TB_VTB_ChargeParameterDiscovery_008();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_015() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
}else {

```

```

        log("PreCondition was unsuccessful.");
    }

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_016() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_017() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_018() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);

```

```

    preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_019() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_020() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_021() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_022() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_023() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
    }else {

```

```

        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_024() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_025() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_026() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_027() runs on SECC_Tester
system SystemSECC {

```



```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
    f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_028() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_029() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_030() runs on SECC_Tester
system SystemSECC {

```

```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
    preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_031() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
    preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_032() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
    preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

```

```

}

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_033() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_034() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
        preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_035() runs on SECC_Tester

```

```

system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
    preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_036() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
    preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ChargeParameterDiscovery_001() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_CMN_TB_VTB_ChargeParameterDiscovery_001();
}else {
    log("PreCondition was unsuccessful.");
}
}

```

```

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ChargeParameterDiscovery_002() runs on SECC_Tester
system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_CMN_TB_VTB_ChargeParameterDiscovery_002();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.1.2.8 SECC test cases for PowerDelivery

```

module TestCases_SECC_PowerDelivery {

import from DataStructure_HAL_61851 all;
import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
import from Pics_15118_2 all;
import from TestBehavior_SECC_PowerDelivery all;
import from ComponentsAndPorts all;
import from Preconditions_SECC_15118_2 all;
import from PostConditions_SECC_15118_2 all;
import from Configurations_15118_2 all;

testcase TC_SECC_AC_VTB_PowerDelivery_001() runs on SECC_Tester system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener, fail);
}
else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_PowerDelivery_002() runs on SECC_Tester system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener, fail);
}
}
}

```

```

else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_PowerDelivery_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_SECC_AC_PR_ChargingStatusOrMeteringReceiptRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_PowerDelivery_001(renegotiate, v_HAL_61851_Listener, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_PowerDelivery_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_SECC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_PowerDelivery_001(start, v_HAL_61851_Listener, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_PowerDelivery_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_PowerDelivery_001(stop, v_HAL_61851_Listener, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_PowerDelivery_006() runs on SECC_Tester system SystemSECC {

```

```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_PowerDelivery_002(start_, v_HAL_61851_Listener);
}
else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_PowerDelivery_007() runs on SECC_Tester system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict :=
f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_PowerDelivery_002(stop_, v_HAL_61851_Listener);
}
else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_PowerDelivery_008() runs on SECC_Tester system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_PowerDelivery_002(start_, v_HAL_61851_Listener);
}
else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_PowerDelivery_009() runs on SECC_Tester system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict :=
f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_PowerDelivery_003(stop_, v_HAL_61851_Listener);
}

```



```

    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_PowerDelivery_010() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_PowerDelivery_004(start_, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_PowerDelivery_011() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_PowerDelivery_005(start_, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_PowerDelivery_012() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_PowerDelivery_001();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_PowerDelivery_013() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;

```

```

var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict :=
f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_CMN_TB_VTB_PowerDelivery_001();
}
else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_PowerDelivery_014() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_001() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        f_SECC_DC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener,
            false, true, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        vc_DC_EVStatus.eVReady := false;
        f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,

```

```

        true, true, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        f_SECC_DC_TB_VTB_PowerDelivery_002();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        vc_DC_EVStatus.evReady := false;
        f_SECC_DC_TB_VTB_PowerDelivery_002();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        f_SECC_DC_TB_VTB_PowerDelivery_003(start_);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

```

```

testcase TC_SECC_DC_VTB_PowerDelivery_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        f_SECC_DC_TB_VTB_PowerDelivery_003(stop_);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_007() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        f_SECC_DC_TB_VTB_PowerDelivery_004(start_);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_008() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        vc_DC_EVStatus.evReady := true;
        f_SECC_DC_TB_VTB_PowerDelivery_005(start_);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_009() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){

```

```

        f_SECC_CMN_TB_VTB_PowerDelivery_001();
    }
    else {
        log("PreCondition was unsuccessful.");
    }
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_010() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        f_SECC_CMN_TB_VTB_PowerDelivery_001();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_011() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_SECC_DC_PR_CurrentDemandOrMeteringReceiptRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_PowerDelivery_001(renegotiate, v_HAL_61851_Listener,
            false, true, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_012() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_SECC_DC_PR_ChargeParameterDiscoveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_PowerDelivery_001(start, v_HAL_61851_Listener,
            true, true, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);

```

```

    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_013() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        f_SECC_DC_TB_VTB_PowerDelivery_006();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_014() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    vc_DC_EVStatus.eVErrorCode := FAILED_RESSTemperatureInhibit;
    preConVerdict :=
    f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        vc_DC_EVStatus.eVReady := false;
        f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                           true, true, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_015() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    vc_DC_EVStatus.eVErrorCode := FAILED_EVShiftPosition;
    preConVerdict :=
    f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        vc_DC_EVStatus.eVReady := false;
        f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                           true, true, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_016() runs on SECC_Tester system SystemSECC {

```

```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
vc_DC_EVStatus.eVErrorCode := FAILED_ChargerConnectorLockFault;
preConVerdict :=
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

//----- Test behavior-----
if ( preConVerdict == pass ){
    vc_DC_EVStatus.eVReady := false;
    f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                     true, true, fail);
}
else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_017() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    vc_DC_EVStatus.eVErrorCode := FAILED_EVRESSMalfunction;
    preConVerdict :=
    f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        vc_DC_EVStatus.eVReady := false;
        f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                           true, true, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_018() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    vc_DC_EVStatus.eVErrorCode := FAILED_ChargingCurrentdifferential;
    preConVerdict :=
    f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        vc_DC_EVStatus.eVReady := false;
        f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                           true, true, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_019() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;

```



```

var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
vc_DC_EVStatus.eVErrorCode := FAILED_ChargingVoltageOutOfRange;
preConVerdict :=
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

//----- Test behavior-----
if ( preConVerdict == pass ){
    vc_DC_EVStatus.eVReady := false;
    f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                     true, true, fail);
}
else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_020() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    vc_DC_EVStatus.eVErrorCode := FAILED_ChargingSystemIncompatibility;
    preConVerdict :=
    f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        vc_DC_EVStatus.eVReady := false;
        f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                           true, true, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_021() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    vc_DC_EVStatus.eVErrorCode := noData;
    preConVerdict :=
    f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ){
        vc_DC_EVStatus.eVReady := false;
        f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                           true, true, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}
}

```

E.1.2.9 SECC test cases for CertificateUpdate

```

module TestCases_SECC_CertificateUpdate {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_CertificateUpdate all;
    import from Services_HAL_61851 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from Templates_CMN_Security all;

    testcase TC_SECC_CMN_VTB_CertificateUpdate_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_CertificateUpdate_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_CertificateUpdate_002() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_CertificateUpdate_002();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_CertificateUpdate_003() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_CertificateUpdate_003();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_CertificateUpdate_004() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
    }
}

```

```

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_CMN_TB_VTB_CertificateUpdate_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_005() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateUpdate_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_006() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateUpdate_006();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_007() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateUpdate_007("WrongSubCertOrder");
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----

```

```

f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_008() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateUpdate_007("WrongPKILeafCert");
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_009() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateUpdate_007("WrongPKISub2Cert");
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_010() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateUpdate_007("WrongPKISub1Cert");
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_011() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

```

```

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_CMN_TB_VTB_CertificateUpdate_007("WrongPKICertChain");
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_012() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateUpdate_008("DE1ABCD2EF357B");
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_013() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateUpdate_008("Invalid_EMAID");
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_014() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateUpdate_009("CRT_CONTRACT_LEAF_WRONG_EMAID");
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_SECC_CMN_VTB_CertificateUpdate_015() runs on SECC_Tester
                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateUpdate_009("CRT_CONTRACT_LEAF_INVALID_EMAID");
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_016() runs on SECC_Tester
                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateUpdate_010("CRT_CONTRACT_LEAF_EXPIRED",
                                                FAILED_CertificateExpired);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_017() runs on SECC_Tester
                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateUpdate_010("CRT_CONTRACT_LEAF_INVALID_CONTENT",
                                                m_CMN_failedResponseCodeList_001());
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

}

```

E.1.2.10 SECC test cases for CertificateInstallation

```

module TestCases_SECC_CertificateInstallation {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_CertificateInstallation all;
    import from Services_HAL_61851 all;

```

```

import from TestBehavior_SECC_CommonBehavior all;
import from PreConditions_SECC_15118_2 all;
import from PostConditions_SECC_15118_2 all;
import from ComponentsAndPorts all;
import from Configurations_15118_2 all;
import from Templates_CMN_Security all;

testcase TC_SECC_CMN_VTB_CertificateInstallation_001() runs on SECC_Tester
                                     system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateInstallation_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateInstallation_002() runs on SECC_Tester
                                     system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateInstallation_002();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateInstallation_003() runs on SECC_Tester
                                     system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateInstallation_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateInstallation_004() runs on SECC_Tester
                                     system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

```



```

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_CMN_TB_VTB_CertificateInstallation_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateInstallation_005() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateInstallation_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateInstallation_006() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateInstallation_006();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateInstallation_007() runs on SECC_Tester
system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateInstallation_007("CRT_OEM_LEAF_EXPIRED",
                                                    FAILED_CertificateExpired);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----

```

```

    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateInstallation_008() runs on SECC_Tester
                                                    system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateInstallation_007("CRT_OEM_LEAF_INVALID_CONTENT",
                                                    m_CMN_failedResponseCodeList_001());
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.1.2.11 SECC test cases for SessionStop

```

module TestCases_SECC_SessionStop {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from TestBehavior_SECC_SessionStop all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_SECC_AC_VTB_SessionStop_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_AC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener, fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_AC_VTB_SessionStop_002() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_AC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_SessionStop_002();
        }else {
            log("PreCondition was unsuccessful.");
        }
    }
}

```

```

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_SessionStop_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_SessionStop_003(terminate);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_SessionStop_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_SessionStop_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionStop_001() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionStop_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

    //----- Test behavior-----

```

```

    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionStop_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_SessionStop_002();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionStop_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_SessionStop_003(terminate);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionStop_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_SessionStop_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionStop_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

```

```

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, false);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_CMN_TB_VTB_SessionStop_005(terminate);
}
else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionStop_007() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_SessionStop_002();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionStop_008() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_SessionStop_003(terminate);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionStop_009() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_SessionStop_004();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

}

E.1.2.12 SECC test cases for ChargingStatus

```

module TestCases_SECC_ChargingStatus {

    import from Pics_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_ChargingStatus all;
    import from Services_HAL_61851 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;

    testcase TC_SECC_AC_VTB_ChargingStatus_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

        //----- Test behavior-----

        if(preConVerdict==pass){
            f_SECC_AC_TB_VTB_ChargingStatus_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }

    testcase TC_SECC_AC_VTB_ChargingStatus_002() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

        //----- Test behavior-----

        if(preConVerdict==pass){
            f_SECC_AC_TB_VTB_ChargingStatus_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }

    testcase TC_SECC_AC_VTB_ChargingStatus_003() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

        //----- Test behavior-----

        if(preConVerdict==pass){
            f_SECC_AC_TB_VTB_ChargingStatus_002();
        }else {
            log("PreCondition was unsuccessful.");
        }
    }
}

```

```

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_ChargingStatus_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----

    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargingStatus_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_ChargingStatus_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----

    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargingStatus_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_ChargingStatus_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----

    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargingStatus_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_ChargingStatus_007() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

```



```

//----- Test behavior-----

if(preConVerdict==pass){
    f_SECC_AC_TB_VTB_ChargingStatus_005();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_ChargingStatus_008()runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----

    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargingStatus_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_ChargingStatus_009()runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----

    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargingStatus_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}
}

```

E.1.2.13 SECC test cases for MeteringReceipt

```

module TestCases_SECC_MeteringReceipt {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_MeteringReceipt all;
    import from Services_HAL_61851 all;
    import from Pics_15118_2 all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;

    testcase TC_SECC_AC_VTB_MeteringReceipt_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
    }
}

```

```

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if ( preConVerdict == pass ) {
    f_SECC_AC_TB_VTB_MeteringReceipt_001(fail);
}
else {
    log ( "PreCondition was unsuccessful." );
}

// ----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_MeteringReceipt_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_CMN_TB_VTB_MeteringReceipt_001();
    }
    else {
        log ("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_SECC_AC_VTB_MeteringReceipt_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_AC_TB_VTB_MeteringReceipt_002();
    }
    else {
        log ("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_SECC_AC_VTB_MeteringReceipt_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
    preConVerdict := f_SECC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_AC_TB_VTB_MeteringReceipt_003();
    }
    else {
        log("PreCondition was unsuccessful.");
    }
}

```

```

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_SECC_AC_VTB_MeteringReceipt_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
    preConVerdict := f_SECC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_CMN_TB_VTB_MeteringReceipt_002();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_SECC_AC_VTB_MeteringReceipt_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
    preConVerdict := f_SECC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_AC_TB_VTB_MeteringReceipt_004();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_SECC_AC_VTB_MeteringReceipt_007() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
    preConVerdict := f_SECC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_AC_TB_VTB_MeteringReceipt_005();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_SECC_DC_VTB_MeteringReceipt_001() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

```

```

// ----- Test behavior-----
if ( preConVerdict == pass ) {
    f_SECC_DC_TB_VTB_MeteringReceipt_001(fail);
}
else {
    log ( "PreCondition was unsuccessful." );
}

// ----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_MeteringReceipt_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_CMN_TB_VTB_MeteringReceipt_001();
    }
    else {
        log ("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_SECC_DC_VTB_MeteringReceipt_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_DC_TB_VTB_MeteringReceipt_002();
    }
    else {
        log ("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_SECC_DC_VTB_MeteringReceipt_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
    preConVerdict := f_SECC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_DC_TB_VTB_MeteringReceipt_003();
    }
    else {
        log("PreCondition was unsuccessful.");
    }
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

```

```

testcase TC_SECC_DC_VTB_MeteringReceipt_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
    preConVerdict := f_SECC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_CMN_TB_VTB_MeteringReceipt_002();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_SECC_DC_VTB_MeteringReceipt_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
    preConVerdict := f_SECC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_DC_TB_VTB_MeteringReceipt_004();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_SECC_DC_VTB_MeteringReceipt_007() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
    preConVerdict := f_SECC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_SECC_DC_TB_VTB_MeteringReceipt_005();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}
}

```

E.1.2.14 SECC test cases for CableCheck

```

module TestCases_SECC_CableCheck {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_Authorization all;
    import from Services_HAL_61851 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

```

```

import from PreConditions_SECC_15118_2 all;
import from TestBehavior_SECC_CableCheck all;
import from PostConditions_SECC_15118_2 all;

testcase TC_SECC_DC_VTB_CableCheck_001() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass){
        f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_CableCheck_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass){
        f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_CableCheck_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass){
        f_SECC_DC_TB_VTB_CableCheck_002(v_HAL_61851_Listener);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_CableCheck_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass){
        f_SECC_DC_TB_VTB_CableCheck_003(v_HAL_61851_Listener);
    }
}

```

```

    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_CableCheck_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass){
        f_SECC_DC_TB_VTB_CableCheck_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_CableCheck_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass){
        f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_CableCheck_007() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict == pass){
        f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_CableCheck_008() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);

```



```

preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict == pass){
    f_SECC_DC_TB_VTB_CableCheck_005();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.1.2.15 SECC test cases for PreCharge

```

module TestCases_SECC_PreCharge {

import from DataStructure_HAL_61851 all;
import from TestBehavior_SECC_Authorization all;
import from Configurations_15118_2 all;
import from Services_HAL_61851 all;
import from TestBehavior_SECC_CableCheck all;
import from PreConditions_SECC_15118_2 all;
import from PostConditions_SECC_15118_2 all;
import from TestBehavior_SECC_PreCharge all;
import from ComponentsAndPorts all;

testcase TC_SECC_DC_VTB_PreCharge_001() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if (preConVerdict == pass) {
        f_SECC_DC_TB_VTB_PreCharge_001(fail);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_PreCharge_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if (preConVerdict == pass) {
        f_SECC_DC_TB_VTB_PreCharge_002();
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_PreCharge_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----

```

```

f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

//----- Test behavior-----
if (preConVerdict == pass) {
    f_SECC_DC_TB_VTB_PreCharge_003();
} else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_PreCharge_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if (preConVerdict == pass) {
        f_SECC_DC_TB_VTB_PreCharge_004();
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.1.2.16 SECC test cases for CurrentDemand

```

module TestCases_SECC_CurrentDemand {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_PowerDelivery all;
    import from TestBehavior_SECC_CurrentDemand all;
    import from TestBehavior_SECC_MeteringReceipt all;
    import from Configurations_15118_2 all;
    import from Services_HAL_61851 all;
    import from TestBehavior_SECC_CableCheck all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;

    testcase TC_SECC_DC_VTB_CurrentDemand_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_SECC_DC_TB_VTB_CurrentDemand_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }

    testcase TC_SECC_DC_VTB_CurrentDemand_002() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
    }
}

```

```

//----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_DC_TB_VTB_CurrentDemand_001(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_CurrentDemand_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_CurrentDemand_002();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_CurrentDemand_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_CurrentDemand_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_CurrentDemand_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_CurrentDemand_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

```

```

testcase TC_SECC_DC_VTB_CurrentDemand_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_CurrentDemand_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_CurrentDemand_007() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_CurrentDemand_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_CurrentDemand_008() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_CurrentDemand_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_CurrentDemand_009() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_CurrentDemand_005();
    }else {
        log("PreCondition was unsuccessful.");
    }
}

```

```

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

```

E.1.2.17 SECC test cases for WeldingDetection

```

module TestCases_SECC_WeldingDetection {

import from DataStructure_HAL_61851 all;
import from Services_HAL_61851 all;
import from TestBehavior_SECC_CommonBehavior all;
import from TestBehavior_SECC_WeldingDetection all;
import from PreConditions_SECC_15118_2 all;
import from PostConditions_SECC_15118_2 all;
import from ComponentsAndPorts all;
import from Configurations_15118_2 all;

testcase TC_SECC_DC_VTB_WeldingDetection_001() runs on SECC_Tester system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_DC_TB_VTB_WeldingDetection_001(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_WeldingDetection_002() runs on SECC_Tester system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_DC_TB_VTB_WeldingDetection_002();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_WeldingDetection_003() runs on SECC_Tester system SystemSECC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

//----- Test behavior-----
if(preConVerdict==pass){
    f_SECC_DC_TB_VTB_WeldingDetection_003();
}else {
    log("PreCondition was unsuccessful.");
}
}
}

```

```

//----- Post Conditions-----
f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_WeldingDetection_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_WeldingDetection_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_WeldingDetection_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, false);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_WeldingDetection_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.2 EVCC test cases

E.2.1 V2G protocol handshake

E.2.1.1 EVCC test cases for SupportedAppProtocol

```

module TestCases_EVCC_SupportedAppProtocol {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Preconditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from Timer_15118_2 all;
    import from TestBehavior_EVCC_SupportedAppProtocol all;

    testcase TC_EVCC_CMN_VTB_SupportedAppProtocol_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_SDP_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){

```

```

        f_EVCC_CMN_PR_InitConfiguration_002(system, fail);
        f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SupportedAppProtocol_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SupportedAppProtocol_002();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SupportedAppProtocol_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SupportedAppProtocol_003();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SupportedAppProtocol_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SDP_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_PR_InitConfiguration_002(system, fail);
        f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```



```

testcase TC_EVCC_CMN_VTB_SupportedAppProtocol_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SDP_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_PR_InitConfiguration_002(system, fail);
        f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.2.2 V2G messages

E.2.2.1 EVCC test cases for SessionSetup

```

module TestCases_EVCC_SessionSetup {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_SessionSetup all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Services_HAL_61851 all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;

    testcase TC_EVCC_CMN_VTB_SessionSetup_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SessionSetup_001(fail);
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_SessionSetup_002() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SessionSetup_002(failed);
        }
        else {
            log("PreCondition was unsuccessful.");
        }
    }
}

```

```

}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SessionSetup_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_002(fAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SessionSetup_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SessionSetup_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SessionSetup_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

```

```

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_SessionSetup_005(e_OscOff);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SessionSetup_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_005(e_NegVolt12);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SessionSetup_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionSetup_001() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }
}

```

```

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionSetup_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionSetup_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }
}

```

```

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionSetup_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionSetup_001() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionSetup_002() runs on EVCC_Tester system SystemEVCC {

```

```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
    v_HAL_61851_Listener);

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionSetup_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionSetup_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----

```

```

f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
    v_HAL_61851_Listener);

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.2.2.2 EVCC test cases for ServiceDiscovery

```

module TestCases_EVCC_ServiceDiscovery {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_ServiceDiscovery all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_ServiceDiscovery all;
    import from Services_HAL_61851 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_CMN_VTB_ServiceDiscovery_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDiscovery_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_ServiceDiscovery_002() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);
    }
}

```



```

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDiscovery_002(fAILED);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDiscovery_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDiscovery_002(fAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDiscovery_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDiscovery_002(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDiscovery_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDiscovery_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDiscovery_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

```

```

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDiscovery_004(e_OscOff);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDiscovery_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDiscovery_004(e_NegVolt12);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDiscovery_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDiscovery_005();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.2.2.3 EVCC test cases for ServiceDetail

```

module TestCases_EVCC_ServiceDetail {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_ServiceDetail all;

    testcase TC_EVCC_CMN_VTB_ServiceDetail_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----

```

```

f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDetail_001(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_002(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_EVCC_CMN_VTB_ServiceDetail_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }
}

```

```

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED_ServiceIDInvalid);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED_ServiceIDInvalid);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_012() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){

```

```

        f_EVCC_CMN_TB_VTB_ServiceDetail_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_013() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_014() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_015() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_006(e_OscOff);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_016() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----

```

```

f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDetail_006(e_NegVolt12);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_017() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_006(e_OscOff);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_018() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_006(e_NegVolt12);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.2.2.4 EVCC test cases for ServiceDetailAndPaymentSelection

```

module TestCases_EVCC_ServiceDetailAndPaymentSelection {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
    import from Timer_15118 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Services_HAL_61851 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;

    testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001() runs on EVCC_Tester
                                                system SystemEVCC {

```



```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_001(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002() runs on EVCC_Tester
                                     system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_002(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_003() runs on EVCC_Tester
                                     system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict :=
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005(failed);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_004() runs on EVCC_Tester
                                     system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict :=
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005(failed_SequenceError);
}else {

```

```

        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_005() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_006() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005(fAILED_PaymentSelectionInvalid);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_007() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005(fAILED_ServiceSelectionInvalid);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_008() runs on EVCC_Tester
                                     system SystemEVCC {

```

```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict :=
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005(fAILED_NoChargeServiceSelected);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_006();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_010() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_007();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_011() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----

```

```

    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_008(e_OscOff);
    }else {
        log("PreCondition was unsuccessful.");
    }
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_012() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_008(e_NegVolt12);
    }else {
        log("PreCondition was unsuccessful.");
    }
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass){
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);

```

```

preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }
}

```

```

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }
}

```

```

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

```



```

}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----

```

```

    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003() runs on EVCC Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004() runs on EVCC Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
    }else {

```

```

        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }
}

```

```

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);

```

```

        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}

```

E.2.2.5 EVCC test cases for PaymentServiceSelection

```

module TestCases_EVCC_PaymentServiceSelection {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_PaymentServiceSelection all;
    import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
    import from Timer_15118 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Services_HAL_61851 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;

    testcase TC_EVCC_CMN_VTB_PaymentServiceSelection_001() runs on EVCC_Tester
                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_003();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_PaymentServiceSelection_002() runs on EVCC_Tester
                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_004();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_PaymentServiceSelection_003() runs on EVCC_Tester
                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_PaymentServiceSelection_001();
        }
    }
}

```

```

    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_PaymentServiceSelection_004() runs on EVCC_Tester
                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_PaymentServiceSelection_002();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_001() runs on EVCC_Tester
                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_PaymentServiceSelection_001(cc_dutyCycle_32A);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_002() runs on EVCC_Tester
                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----

```

```

    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
    }else {
        log("PreCondition was unsuccessful.");
    }
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_003() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_004() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
    }else {
        log("PreCondition was unsuccessful.");
    }
}

```



```

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_005() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_006() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_007() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_008() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_009() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;

```

```

var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_010() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_011() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

```

```

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_012() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_013() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }
}

```

```

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_014() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_015() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }
}

```

```

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_016() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_017() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }
}

```

```

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_001() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_002() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
    }
}

```



```

    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_003() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_004() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
    }else {
        log("PreCondition was unsuccessful.");
    }
}

```

```

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_005() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
v_HAL_61851_Listener);

if(preConVerdict == pass) {
preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
f_EVCC_startSleepingPhase(PixIT_CMN_CMN_WakeUp - 5.0);
}

if(preConVerdict == pass) {
unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
preConVerdict :=
f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
}else {
log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_006() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
v_HAL_61851_Listener);

if(preConVerdict == pass) {
preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
f_EVCC_startSleepingPhase(PixIT_CMN_CMN_WakeUp - 5.0);
}

if(preConVerdict == pass) {
unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
preConVerdict :=
f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
}else {
log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);

```

```

    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_007() runs on EVCC_Tester
                                                system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_008() runs on EVCC_Tester
                                                system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_009() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_010() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass){
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_011() runs on EVCC_Tester
system SystemEVCC {

```

```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
    v_HAL_61851_Listener);

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_012() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_013() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

```

```

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
    v_HAL_61851_Listener);

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_014() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_015() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(

```

```

        v_HAL_61851_Listener);

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_016() runs on EVCC Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.2.2.6 EVCC test cases for CertificateInstallation

```

module TestCases_EVCC_CertificateInstallation {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from TestBehavior_EVCC_CertificateInstallation all;
    import from PreConditions_EVCC_15118_2 all;

```



```

import from PostConditions_EVCC_15118_2 all;
import from ComponentsAndPorts all;
import from Configurations_15118_2 all;

testcase TC_EVCC_CMN_VTB_CertificateInstallation_001() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateInstallation_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateInstallation_002() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateInstallation_002(failed);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateInstallation_003() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateInstallation_002(failed_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateInstallation_004() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);

```

```

preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
  f_EVCC_CMN_TB_VTB_CertificateInstallation_002(fAILED_UnknownSession);
}else {
  log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateInstallation_005() runs on EVCC_Tester
system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // ----- Pre Conditions-----
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

  //----- Test behavior-----
  if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_CertificateInstallation_002(fAILED_SignatureError);
  }else {
    log("PreCondition was unsuccessful.");
  }

  //----- Post Conditions-----
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateInstallation_006() runs on EVCC_Tester
system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // ----- Pre Conditions-----
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

  //----- Test behavior-----
  if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_CertificateInstallation_002(fAILED_CertificateExpired);
  }else {
    log("PreCondition was unsuccessful.");
  }

  //----- Post Conditions-----
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateInstallation_007() runs on EVCC_Tester
system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // ----- Pre Conditions-----
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

  //----- Test behavior-----
  if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_CertificateInstallation_002(fAILED_CertificateRevoked);
  }else {
    log("PreCondition was unsuccessful.");
  }

  //----- Post Conditions-----
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_EVCC_CMN_VTB_CertificateInstallation_008() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateInstallation_002(fAILED_NoCertificateAvailable);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateInstallation_009() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateInstallation_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateInstallation_010() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateInstallation_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateInstallation_011() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){

```

```

        f_EVCC_CMN_TB_VTB_CertificateInstallation_005(e_OscOff);
    }else {
        log("PreCondition was unsuccessful.");
    }
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateInstallation_012() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateInstallation_005(e_NegVolt12);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.2.2.7 EVCC test cases for CertificateUpdate

```

module TestCases_EVCC_CertificateUpdate {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from TestBehavior_EVCC_CertificateUpdate all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_CMN_VTB_CertificateUpdate_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict :=
        f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_CertificateUpdate_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_CertificateUpdate_002() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);
    }
}

```

```

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateUpdate_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateUpdate_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateUpdate_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_SignatureError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateUpdate_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;

```

```

var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
  f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_CertificateExpired);
}else {
  log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateUpdate_007() runs on EVCC_Tester system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // ----- Pre Conditions-----
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

  //----- Test behavior-----
  if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_CertificateRevoked);
  }else {
    log("PreCondition was unsuccessful.");
  }

  //----- Post Conditions-----
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateUpdate_008() runs on EVCC_Tester system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // ----- Pre Conditions-----
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

  //----- Test behavior-----
  if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_NoCertificateAvailable);
  }else {
    log("PreCondition was unsuccessful.");
  }

  //----- Post Conditions-----
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateUpdate_009() runs on EVCC_Tester system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // ----- Pre Conditions-----
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

  //----- Test behavior-----
  if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_ContractCanceled);
  }else {
    log("PreCondition was unsuccessful.");
  }

  //----- Post Conditions-----
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

}

testcase TC_EVCC_CMN_VTB_CertificateUpdate_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_ChallengeInvalid);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateUpdate_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateUpdate_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateUpdate_012() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateUpdate_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateUpdate_013() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateUpdate_005(e_OscOff);
    }else {
        log("PreCondition was unsuccessful.");
    }
}

```



```

    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateUpdate_014() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateUpdate_005(e_NegVolt12);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.2.2.8 EVCC test cases for CertificateHandlingAndPaymentDetails

```

module TestCases_EVCC_CertificateHandlingAndPaymentDetails {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from TestBehavior_EVCC_CertificateHandlingAndPaymentDetails all;
    import from Preconditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001() runs on EVCC_Tester
                                                                    system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict :=
        f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_002() runs on EVCC_Tester
                                                                    system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict :=
        f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

        //----- Test behavior-----

```

```

    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_002(fAILED);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_003() runs on EVCC_Tester
                                                system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_002(fAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_004() runs on EVCC_Tester
                                                system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_002(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005() runs on EVCC_Tester
                                                system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_006() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_007() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_005(e_OscOff);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_008() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_005(e_NegVolt12);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.2.2.9 EVCC test cases for PaymentDetails

```

module TestCases_EVCC_PaymentDetails {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851 all;

```

```

import from TestBehavior_EVCC_CommonBehavior all;
import from TestBehavior_EVCC_PaymentDetails all;
import from PreConditions_EVCC_15118_2 all;
import from PostConditions_EVCC_15118_2 all;
import from ComponentsAndPorts all;
import from Configurations_15118_2 all;

testcase TC_EVCC_CMN_VTB_PaymentDetails_001() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_PaymentDetails_001();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_PaymentDetails_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_PaymentDetails_002();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_PaymentDetails_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_PaymentDetails_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.2.2.10 EVCC test cases for Authorization

```

module TestCases_EVCC_Authorization {

    import from DataStructure_HAL_61851 all;

```

```

import from TestBehavior_EVCC_Authorization all;
import from Services_HAL_61851 all;
import from TestBehavior_EVCC_CommonBehavior all;
import from PreConditions_EVCC_15118_2 all;
import from PostConditions_EVCC_15118_2 all;
import from ComponentsAndPorts all;
import from Configurations_15118_2 all;

testcase TC_EVCC_CMN_VTB_Authorization_001() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_002(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_003(fAILED);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----

```

```

f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_Authorization_003(fAILED_SequenceError);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_003(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_003(fAILED_ChallengeInvalid);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_003(fAILED_SignatureError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_008() runs on EVCC_Tester system SystemEVCC {

```

```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_Authorization_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }
}

```



```

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_012() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_006();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_013() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_007(e_OscOff);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_014() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_007(e_NegVolt12);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.2.2.11 EVCC test cases for ChargeParameterDiscovery

```

module TestCases_EVCC_ChargeParameterDiscovery {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_ChargeParameterDiscovery all;
    import from Services_HAL_61851 all;

```

```

import from TestBehavior_EVCC_CommonBehavior all;
import from PreConditions_EVCC_15118_2 all;
import from PostConditions_EVCC_15118_2 all;
import from ComponentsAndPorts all;
import from Configurations_15118_2 all;
import from Timer_15118 all;
import from Pics_15118 all;
import from Pixit_15118 all;
import from Timer_15118_2 all;

```

```

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_001() runs on EVCC_Tester
                                                    system SystemEVCC {

```

```

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_002() runs on EVCC_Tester
                                                    system SystemEVCC {

```

```

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001(fAILED);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_003() runs on EVCC_Tester
                                                    system SystemEVCC {

```

```

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001(fAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_004() runs on EVCC_Tester

```

```

system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001(FAILED_UnknownSession);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_005() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001(FAILED_WrongEnergyTransferMode);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_006() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001(FAILED_WrongChargeParameter);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_007() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----

```

```

if(preConVerdict==pass){
    f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_002();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_008() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_003(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_009() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_004(e_OscOff);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_010() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_004(e_NegVolt12);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_011() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_003(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_012() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_013() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_006();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_014() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

```

```

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_006();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_015() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_016() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
    }else {

```

```

        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_017() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_018() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);

```



```

    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_019() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_020() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_021() runs on EVCC_Tester
system SystemEVCC {

```

```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_022() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_023() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----

```

```

f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_024() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_025() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

```

```

    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_026() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_001() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }
}

```

```

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_002() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001(fAILED);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_003() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001(fAILED_SequenceError);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_004() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001(fAILED_UnknownSession);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_005() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

```

```

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001(FAILED_WrongEnergyTransferMode);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_006() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001(FAILED_WrongChargeParameter);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_007() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_002();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_008() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_003(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }
}

```

```

    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_009() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_004(e_OscOff);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_010() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_004(e_NegVolt12);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_011() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_003(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_012() runs on EVCC_Tester
system SystemEVCC {

```



```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_005();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_013() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict :=
f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_006();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_014() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict :=
f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_006();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_015() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
    v_HAL_61851_Listener);

if(preConVerdict == pass) {
    preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
}

```

```

if(preConVerdict == pass) {
    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
    f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_016() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass){
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_017() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);

```

```

    f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
}

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_018() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_019() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }
}

```

```

if(preConVerdict == pass) {
    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    preConVerdict :=
        f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_020() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(PXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_021() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=

```

```

    f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
}

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_022() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_023() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }
}

```

```

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_024() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_025() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
    }
}

```

```

    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_026() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
        v_HAL_61851_Listener);

    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        f_EVCC_startSleepingPhase(par_SECC_Pmax0W - 5.0);
    }

    if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_002();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ChargeParameterDiscovery_002() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

```



```

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_003();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.2.2.12 EVCC test cases for PowerDelivery

```

module TestCases_EVCC_PowerDelivery {

import from DataStructure_HAL_61851 all;
import from TestBehavior_EVCC_PowerDelivery all;
import from Services_HAL_61851 all;
import from TestBehavior_EVCC_CommonBehavior all;
import from PreConditions_EVCC_15118_2 all;
import from PostConditions_EVCC_15118_2 all;
import from ComponentsAndPorts all;
import from Configurations_15118_2 all;

testcase TC_EVCC_AC_VTB_PowerDelivery_001() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_001(v_HAL_61851_Listener, fail);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_002(fAILED, v_HAL_61851_Listener);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

```

```

//----- Test behavior-----
if( preConVerdict == pass ) {
    f_EVCC_AC_TB_VTB_PowerDelivery_002(fAILED_SequenceError, v_HAL_61851_Listener);
} else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_002(fAILED_UnknownSession, v_HAL_61851_Listener);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_002(fAILED_ChargingProfileInvalid, v_HAL_61851_Listener);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_002(fAILED_TariffSelectionInvalid, v_HAL_61851_Listener);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;

```

```

var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

//----- Test behavior-----
if( preConVerdict == pass ) {
    f_EVCC_AC_TB_VTB_PowerDelivery_002(fAILED_PowerDeliveryNotApplied, v_HAL_61851_Listener);
} else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_002(fAILED_ContactorError, v_HAL_61851_Listener);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_003(v_HAL_61851_Listener);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_004(v_HAL_61851_Listener);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

```

```

}

testcase TC_EVCC_AC_VTB_PowerDelivery_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_005(e_OscOff);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_012() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_005(e_NegVolt12);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_013() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
        v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006();
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_014() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
        v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006();
    }

```

```

    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_015() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_001(v_HAL_61851_Listener, fail);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_016() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
        v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006();
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_017() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
        v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006();
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_018() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

```

```

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if( preConVerdict == pass ) {
    f_EVCC_AC_TB_VTB_PowerDelivery_001(v_HAL_61851_Listener, fail);
} else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_019() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_001(v_HAL_61851_Listener, fail);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_020() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_001(v_HAL_61851_Listener, fail);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_021() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_001(v_HAL_61851_Listener, fail);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

```

```

testcase TC_EVCC_AC_VTB_PowerDelivery_022() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                                v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006();
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_023() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                                v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006();
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_024() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                                v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006();
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_025() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                                v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006();
    }

```



```

    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_026() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
        v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006();
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_027() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
        v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006();
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_028() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
        v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006();
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_029() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

```

```

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                    v_HAL_61851_Listener);

//----- Test behavior-----
if( preConVerdict == pass ) {
    f_EVCC_AC_TB_VTB_PowerDelivery_006();
} else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_030() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                    v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006();
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_031() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                    v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006();
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_032() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                    v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006();
    } else {
        log("PreCondition was unsuccessful.");
    }
}

```

```

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_033() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
        v_HAL_61851_Listener);

    //----- Test behavior-----
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006();
    } else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}
}

```

E.2.2.13 EVCC test cases for SessionStop

```

module TestCases_EVCC_SessionStop {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_SessionStop all;
    import from TestBehavior_EVCC_WeldingDetectionOrSessionStop all;
    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Preconditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_AC_VTB_SessionStop_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_EVCC_AC_TB_VTB_SessionStop_001(pass, pass, fail);
        }else{
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_AC_VTB_SessionStop_002() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_EVCC_AC_TB_VTB_SessionStop_003(FAILED);
        }else{
            log("PreCondition was unsuccessful.");
        }
    }
}

```

```

    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_003(FAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_003(FAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

```

```

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_AC_TB_VTB_SessionStop_005();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_001(pass, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_001(pass, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_001(pass, inconc, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

```

```

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_AC_TB_VTB_SessionStop_001(pass, inconc, fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_001(pass, inconc, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_012() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_001(inconc, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_013() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_001(inconc, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_EVCC_AC_VTB_SessionStop_014() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_001(inconc, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_015() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_001(inconc, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_016() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_001(inconc, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_017() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_001(inconc, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }
}

```



```

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_018() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_001(inconc, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_019() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_001(inconc, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_020() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_001(inconc, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_001() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----

```

```

    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
            pass, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener, pass, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

```

```

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener, pass, pass);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
            pass, inconc);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, inconc);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

}

testcase TC_EVCC_DC_VTB_SessionStop_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                            pass, inconc);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, inconc);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                            pass, inconc);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_012() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, inconc);

```

```

    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_013() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                            inconc, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_014() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_015() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                            inconc, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_016() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

```

```

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_017() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                            inconc, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_018() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_019() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                            inconc, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

}

testcase TC_EVCC_DC_VTB_SessionStop_020() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_021() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                            inconc, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_022() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_023() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                            inconc, pass);
    }

```



```

    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_024() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_025() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                             inconc, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_026() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_027() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----

```

```

f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                    inconc, pass);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_028() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_029() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                    inconc, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_030() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

    }
}

```

E.2.2.14 EVCC test cases for ChargingStatus

```

module TestCases_EVCC_ChargingStatus {

    import from Pics_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_ChargingStatus all;
    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_AC_VTB_ChargingStatus_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

        //----- Test behavior -----
        if ( preConVerdict == pass ) {
            f_EVCC_AC_TB_VTB_ChargingStatus_001(fail);
        }
        else {
            log ( "PreCondition was unsuccessful." );
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_AC_VTB_ChargingStatus_002() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if ( preConVerdict == pass ) {
            f_EVCC_AC_TB_VTB_ChargingStatus_002(FAILED, v_HAL_61851_Listener);
        }
        else {
            log ( "PreCondition was unsuccessful." );
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_AC_VTB_ChargingStatus_003() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if ( preConVerdict == pass ) {
            f_EVCC_AC_TB_VTB_ChargingStatus_002(FAILED_SequenceError, v_HAL_61851_Listener);
        }
        else {
            log ( "PreCondition was unsuccessful." );
        }
    }
}

```

```

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingStatus_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingStatus_002(fAILED_UnknownSession, v_HAL_61851_Listener);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingStatus_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingStatus_003(v_HAL_61851_Listener);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingStatus_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingStatus_004(e_OscOff);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingStatus_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);

```

```

preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

//----- Test behavior-----
if ( preConVerdict == pass ) {
    f_EVCC_AC_TB_VTB_ChargingStatus_004(e_NegVolt12);
}
else {
    log ( "PreCondition was unsuccessful." );
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingStatus_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingStatus_005(v_HAL_61851_Listener);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}
}

```

E.2.2.15 EVCC test cases for ChargingLoopAcOrPowerDelivery

```

module TestCases_EVCC_ChargingLoopAcOrPowerDelivery {

    import from Pics_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_ChargingLoopAcOrPowerDelivery all;
    import from Services_HAL_61851 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001() runs on EVCC_Tester
                                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

        // ----- Test behavior-----
        if ( preConVerdict == pass ) {
            f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener, fail);
        }
        else {
            log ( "PreCondition was unsuccessful." );
        }

        // ----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002() runs on EVCC_Tester
                                                                system SystemEVCC {

```

```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if ( preConVerdict == pass ) {
    f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_002(v_HAL_61851_Listener, fail);
}
else {
    log ( "PreCondition was unsuccessful." );
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_003() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003(fAILED);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_004() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003(fAILED_SequenceError);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_005() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003(fAILED_UnknownSession);
    }

```

```

    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_006() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003(fAILED_ContactorError);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_004();
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_005(fail);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

```



```

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_006(fail);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_010() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_007();
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_011() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_008(e_OscOff);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_012() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

```

```

// ----- Test behavior-----
if ( preConVerdict == pass ) {
    f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_008(e_NegVolt12);
}
else {
    log ( "PreCondition was unsuccessful." );
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_009(fail);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_010(fail);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}
}

```

E.2.2.16 EVCC test cases for MeteringReceipt

```

module TestCases_EVCC_MeteringReceipt {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_MeteringReceipt all;
    import from Services_HAL_61851 all;
    import from Pics_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_AC_VTB_MeteringReceipt_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
    }
}

```

```

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if (preConVerdict == pass) {
f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
}
else {
log("PreCondition was unsuccessful.");
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_002() runs on EVCC_Tester system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if (preConVerdict == pass) {
f_EVCC_AC_TB_VTB_MeteringReceipt_002(fAILED, v_HAL_61851_Listener);
}
else {
log("PreCondition was unsuccessful.");
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_003() runs on EVCC_Tester system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if (preConVerdict == pass) {
f_EVCC_AC_TB_VTB_MeteringReceipt_002(fAILED_SequenceError, v_HAL_61851_Listener);
}
else {
log("PreCondition was unsuccessful.");
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_004() runs on EVCC_Tester system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if (preConVerdict == pass) {
f_EVCC_AC_TB_VTB_MeteringReceipt_002(fAILED_UnknownSession, v_HAL_61851_Listener);
}
else {
log("PreCondition was unsuccessful.");
}
}

```

```

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_002(fAILED_MeteringSignatureNotValid,
                                             v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_002(fAILED_SignatureError, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_MeteringReceipt_001(v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);

```

```

preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if (preConVerdict == pass) {
    f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
}
else {
    log("PreCondition was unsuccessful.");
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_MeteringReceipt_002(v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_003(e_OscOff);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

```

```

}

testcase TC_EVCC_AC_VTB_MeteringReceipt_012() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_003(e_NegVolt12);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_013() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_014() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_015() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {

```

```

        f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_016() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_017() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_018() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_019() runs on EVCC_Tester system SystemEVCC {

```



```

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if (preConVerdict == pass) {
f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
}
else {
log("PreCondition was unsuccessful.");
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_020() runs on EVCC_Tester system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if (preConVerdict == pass) {
f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
}
else {
log("PreCondition was unsuccessful.");
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_021() runs on EVCC_Tester system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if (preConVerdict == pass) {
f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
}
else {
log("PreCondition was unsuccessful.");
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_001() runs on EVCC_Tester system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if (preConVerdict == pass) {
f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
}
else {
log("PreCondition was unsuccessful.");
}

```

```

    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_MeteringReceipt_002(fAILED, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_MeteringReceipt_002(fAILED_SequenceError, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_MeteringReceipt_002(fAILED_UnknownSession, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----

```

```

f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if (preConVerdict == pass) {
    f_EVCC_DC_TB_VTB_MeteringReceipt_002(fAILED_MeteringSignatureNotValid,
        v_HAL_61851_Listener);
}
else {
    log("PreCondition was unsuccessful.");
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_MeteringReceipt_002(fAILED_SignatureError, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_MeteringReceipt_001(v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----

```

```

f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_MeteringReceipt_002(v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_MeteringReceipt_003(e_OscOff);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_012() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

```

```

// ----- Test behavior-----
if (preConVerdict == pass) {
    f_EVCC_DC_TB_VTB_MeteringReceipt_003(e_NegVolt12);
}
else {
    log("PreCondition was unsuccessful.");
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_013() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_014() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_015() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

```

testcase TC_EVCC_DC_VTB_MeteringReceipt_016() runs on EVCC_Tester system SystemEVCC {

```
var HAL_61851_Listener v_HAL_61851_Listener;
var verdictctype preConVerdict;
```

```
// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);
```

```
// ----- Test behavior-----
if (preConVerdict == pass) {
    f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
}
else {
    log("PreCondition was unsuccessful.");
}
```

```
// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}
```

testcase TC_EVCC_DC_VTB_MeteringReceipt_017() runs on EVCC_Tester system SystemEVCC {

```
var HAL_61851_Listener v_HAL_61851_Listener;
var verdictctype preConVerdict;
```

```
// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);
```

```
// ----- Test behavior-----
if (preConVerdict == pass) {
    f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
}
else {
    log("PreCondition was unsuccessful.");
}
```

```
// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}
```

testcase TC_EVCC_DC_VTB_MeteringReceipt_018() runs on EVCC_Tester system SystemEVCC {

```
var HAL_61851_Listener v_HAL_61851_Listener;
var verdictctype preConVerdict;
```

```
// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);
```

```
// ----- Test behavior-----
if (preConVerdict == pass) {
    f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
}
else {
    log("PreCondition was unsuccessful.");
}
```

```
// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}
```

testcase TC_EVCC_DC_VTB_MeteringReceipt_019() runs on EVCC_Tester system SystemEVCC {

```
var HAL_61851_Listener v_HAL_61851_Listener;
var verdictctype preConVerdict;
```

```
// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);
```

```
// ----- Test behavior-----
if (preConVerdict == pass) {
    f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
}
```

```

else {
    log("PreCondition was unsuccessful.");
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_020() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_021() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}
}

```

E.2.2.17 EVCC test cases for CableCheck

```

module TestCases_EVCC_CableCheck {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_V2GTPSDP all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PostConditions_EVCC_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_CableCheck all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_DC_VTB_CableCheck_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);
    }
}

```



```

//----- Test behavior-----
if (preConVerdict == pass) {
    f_EVCC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, fail);
}
else{
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_CableCheck_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_CableCheck_002(FAILED, v_HAL_61851_Listener);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_CableCheck_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_CableCheck_002(FAILED_SequenceError, v_HAL_61851_Listener);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_CableCheck_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_CableCheck_002(FAILED_UnknownSession, v_HAL_61851_Listener);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

```

```

testcase TC_EVCC_DC_VTB_CableCheck_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_CableCheck_003(v_HAL_61851_Listener);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_CableCheck_006() runs on EVCC_Tester system SystemEVCC{

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_CableCheck_004(v_HAL_61851_Listener);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_CableCheck_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_CableCheck_005(v_HAL_61851_Listener);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_CableCheck_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_CableCheck_006(e_OscOff);
    }
}

```

```

else{
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_CableCheck_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_CableCheck_006(e_NegVolt12);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.2.2.18 EVCC test cases for PreCharge

```

module TestCases_EVCC_PreCharge{

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_V2GTPSDF all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PostConditions_EVCC_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_PreCharge all;
    import from TestCases_EVCC_CableCheck all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_DC_VTB_PreCharge_001() runs on EVCC_Tester system SystemEVCC {

        var verdicttype preConVerdict;
        var HAL_61851_Listener v_HAL_61851_Listener;

        //----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if (preConVerdict == pass) {
            f_EVCC_DC_TB_VTB_PreCharge_001(fail);
        }
        else{
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_PreCharge_002() runs on EVCC_Tester system SystemEVCC {

        var verdicttype preConVerdict;
        var HAL_61851_Listener v_HAL_61851_Listener;

        //----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    }
}

```

```

preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

//----- Test behavior-----

if (preConVerdict == pass) {
    f_EVCC_DC_TB_VTB_PreCharge_002(fAILED, v_HAL_61851_Listener);
}
else{
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PreCharge_003() runs on EVCC_Tester system SystemEVCC {

    var verdicttype preConVerdict;
    var HAL_61851_Listener v_HAL_61851_Listener;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //----- Test behavior-----

    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_PreCharge_002(fAILED_SequenceError, v_HAL_61851_Listener);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PreCharge_004() runs on EVCC_Tester system SystemEVCC {

    var verdicttype preConVerdict;
    var HAL_61851_Listener v_HAL_61851_Listener;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //----- Test behavior-----

    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_PreCharge_002(fAILED_UnknownSession, v_HAL_61851_Listener);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PreCharge_005() runs on EVCC_Tester system SystemEVCC {

    var verdicttype preConVerdict;
    var HAL_61851_Listener v_HAL_61851_Listener;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //----- Test behavior-----

    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_PreCharge_003(v_HAL_61851_Listener);
    }
    else{
        log("PreCondition was unsuccessful.");
    }
}

```

```

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PreCharge_006() runs on EVCC_Tester system SystemEVCC {

    var verdicttype preConVerdict;
    var HAL_61851_Listener v_HAL_61851_Listener;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //----- Test behavior-----

    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_PreCharge_004(v_HAL_61851_Listener);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PreCharge_007() runs on EVCC_Tester system SystemEVCC {

    var verdicttype preConVerdict;
    var HAL_61851_Listener v_HAL_61851_Listener;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //----- Test behavior-----

    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_PreCharge_005(v_HAL_61851_Listener);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PreCharge_008() runs on EVCC_Tester system SystemEVCC {

    var verdicttype preConVerdict;
    var HAL_61851_Listener v_HAL_61851_Listener;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //----- Test behavior-----

    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_PreCharge_006(e_OscOff);
    }
    else{
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PreCharge_009() runs on EVCC_Tester system SystemEVCC {

    var verdicttype preConVerdict;
    var HAL_61851_Listener v_HAL_61851_Listener;

```

```

//----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

//----- Test behavior-----

if (preConVerdict == pass) {
    f_EVCC_DC_TB_VTB_PreCharge_006(e_NegVolt12);
}
else{
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```

E.2.2.19 EVCC test cases for PreChargeOrPowerDelivery

```

module TestCases_EVCC_PreChargeOrPowerDelivery {

    import from Pics_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_PreChargeOrPowerDelivery all;
    import from Services_HAL_61851 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001() runs on EVCC_Tester
                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if ( preConVerdict == pass ) {
            f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_001(fail);
        }
        else {
            log ( "PreCondition was unsuccessful." );
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_002() runs on EVCC_Tester
                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

        // ----- Test behavior-----
        if ( preConVerdict == pass ) {
            f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002(fAILED, v_HAL_61851_Listener);
        }
        else {
            log ( "PreCondition was unsuccessful." );
        }

        // ----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    }
}

```

```

    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_003() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002(fAILED_SequenceError,
                                                    v_HAL_61851_Listener);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_004() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002(fAILED_UnknownSession,
                                                    v_HAL_61851_Listener);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_005() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002(fAILED_ChargingProfileInvalid,
                                                    v_HAL_61851_Listener);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_006() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;

```



```

var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if ( preConVerdict == pass ) {
    f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002(fAILED_TariffSelectionInvalid,
                                                v_HAL_61851_Listener);
}
else {
    log ( "PreCondition was unsuccessful." );
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_007() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if ( preConVerdict == pass ) {
    f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002(fAILED_PowerDeliveryNotApplied,
                                                v_HAL_61851_Listener);
}
else {
    log ( "PreCondition was unsuccessful." );
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_008() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if ( preConVerdict == pass ) {
    f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002(fAILED_ContactorError,
                                                v_HAL_61851_Listener);
}
else {
    log ( "PreCondition was unsuccessful." );
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

// ----- Test behavior-----

```

```

    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_003(v_HAL_61851_Listener);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_010() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_004(v_HAL_61851_Listener);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_011() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_005(e_OscOff);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_012() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_005(e_NegVolt12);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

```

```

}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013() runs on EVCC_Tester
                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014() runs on EVCC_Tester
                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015() runs on EVCC_Tester
                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_001(fail);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016() runs on EVCC_Tester
                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);

```

```

preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

//----- Test behavior-----
if ( preConVerdict == pass ) {
    f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_001(fail);
}
else {
    log ( "PreCondition was unsuccessful." );
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }
}

```

```

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

//----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);

//----- Test behavior-----
if ( preConVerdict == pass ) {
    f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
}
else {
    log ( "PreCondition was unsuccessful." );
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

//----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);

//----- Test behavior-----
if ( preConVerdict == pass ) {
    f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
}
else {
    log ( "PreCondition was unsuccessful." );
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

//----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);

//----- Test behavior-----
if ( preConVerdict == pass ) {
    f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
}
else {
    log ( "PreCondition was unsuccessful." );
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}
}

```

E.2.2.20 EVCC test cases for CurrentDemand

```

module TestCases_EVCC_CurrentDemand {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_PowerDelivery all;
    import from TestBehavior_EVCC_CurrentDemand all;
    import from TestBehavior_EVCC_MeteringReceipt all;
    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_DC_VTB_CurrentDemand_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

        //----- Test behavior -----
        if ( preConVerdict == pass ) {
            f_EVCC_DC_TB_VTB_CurrentDemand_001(fail);
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_DC_VTB_CurrentDemand_002() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

        //----- Test behavior -----
        if ( preConVerdict == pass ) {
            f_EVCC_DC_TB_VTB_CurrentDemand_002(fAILED, v_HAL_61851_Listener);
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_DC_VTB_CurrentDemand_003() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

        //----- Test behavior -----
        if ( preConVerdict == pass ) {
            f_EVCC_DC_TB_VTB_CurrentDemand_002(fAILED_SequenceError, v_HAL_61851_Listener);
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }
}

```

```

}

testcase TC_EVCC_DC_VTB_CurrentDemand_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //----- Test behavior -----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_CurrentDemand_002(fAILED_UnknownSession, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_CurrentDemand_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //----- Test behavior -----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_CurrentDemand_003(v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_CurrentDemand_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //----- Test behavior -----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_CurrentDemand_004(e_OscOff);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_CurrentDemand_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //----- Test behavior -----
    if ( preConVerdict == pass ) {

```



```

        f_EVCC_DC_TB_VTB_CurrentDemand_004(e_NegVolt12);
    }
    else {
        log("PreCondition was unsuccessful.");
    }
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_CurrentDemand_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //----- Test behavior -----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_CurrentDemand_005(v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}
}

```

E.2.2.21 EVCC test cases for ChargingLoopDcOrPowerDelivery

```

module TestCases_EVCC_ChargingLoopDcOrPowerDelivery {

    import from Pics_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_ChargingLoopDcOrPowerDelivery all;
    import from Services_HAL_61851 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001() runs on EVCC_Tester
                                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

        // ----- Test behavior-----
        if ( preConVerdict == pass ) {
            f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_001(fail);
        }
        else {
            log ( "PreCondition was unsuccessful." );
        }

        // ----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002() runs on EVCC_Tester
                                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----

```

```

f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if ( preConVerdict == pass ) {
    f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_002(fail);
}
else {
    log ( "PreCondition was unsuccessful." );
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_003() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003(fAILED, v_HAL_61851_Listener);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_004() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003(fAILED_SequenceError,
            v_HAL_61851_Listener);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_005() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003(fAILED_UnknownSession,
            v_HAL_61851_Listener);
    }
    else {

```

```

    log ( "PreCondition was unsuccessful." );
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_006() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003(fAILED_ContactorError,
                                                         v_HAL_61851_Listener);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_004(v_HAL_61851_Listener);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_005(fail);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009() runs on EVCC_Tester

```

```

system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_006(fail);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_010() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_007(v_HAL_61851_Listener);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_011() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_008(e_OscOff);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_012() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----

```

```

    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_008(e_NegVolt12);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_009(fail);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014() runs on EVCC_Tester
system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_010(fail);
    }
    else {
        log ( "PreCondition was unsuccessful." );
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}
}

```

E.2.2.22 EVCC test cases for WeldingDetection

```

module TestCases_EVCC_WeldingDetection {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_WeldingDetection all;
    import from Services_HAL_61851 all;
    import from Pics_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_DC_VTB_WeldingDetection_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
    }
}

```

```

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

// ----- Test behavior-----
if (preConVerdict == pass) {
    f_EVCC_DC_TB_VTB_WeldingDetection_001(v_HAL_61851_Listener, fail);
}
else {
    log("PreCondition was unsuccessful.");
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_WeldingDetection_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_WeldingDetection_002(fAILED);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_WeldingDetection_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_WeldingDetection_002(fAILED_SequenceError);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_WeldingDetection_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_WeldingDetection_002(fAILED_UnknownSession);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----

```

```

    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_WeldingDetection_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_WeldingDetection_003();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_WeldingDetection_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_WeldingDetection_004();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_WeldingDetection_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_WeldingDetection_005(e_OscOff);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_WeldingDetection_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

```



```

// ----- Test behavior-----
if (preConVerdict == pass) {
    f_EVCC_DC_TB_VTB_WeldingDetection_005(e_NegVolt12);
}
else {
    log("PreCondition was unsuccessful.");
}

// ----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_WeldingDetection_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    // ----- Test behavior-----
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_WeldingDetection_006(v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}
}

```

E.2.2.23 EVCC test cases for WeldingDetectionOrSessionStop

```

module TestCases_EVCC_WeldingDetectionOrSessionStop {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_WeldingDetectionOrSessionStop all;
    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001() runs on EVCC_Tester
                                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ----- Pre Conditions-----
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

        //----- Test behavior-----
        if(preConVerdict==pass){
            f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                pass, pass, fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //----- Post Conditions-----
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_002() runs on EVCC_Tester
                                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
    }
}

```

```

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_004(fAILED);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_003() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_004(fAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_004() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_004(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----

```

```

f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_006() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_006();
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
pass, pass, fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
pass, pass, fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----

```

```

f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                         pass, inconc, fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                             pass, inconc, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                             pass, inconc, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                             inconc, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }
}

```

```

    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                         inconc, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                         inconc, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015() runs on EVCC_Tester
                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                         inconc, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016() runs on EVCC_Tester

```

```

system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                        inconc, pass, fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                        inconc, pass, fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                        inconc, pass, fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019() runs on EVCC_Tester
system SystemEVCC {

var HAL_61851_Listener v_HAL_61851_Listener;
var verdicttype preConVerdict;

// ----- Pre Conditions-----
f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

```

```

//----- Test behavior-----
if(preConVerdict==pass){
    f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                    inconc, pass, fail);
}else {
    log("PreCondition was unsuccessful.");
}

//----- Post Conditions-----
f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020() runs on EVCC_Tester
                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ----- Pre Conditions-----
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //----- Test behavior-----
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                            inconc, pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //----- Post Conditions-----
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}

```


Annex F (normative)

Function specifications for supporting test execution

This annex covers all pre-condition, post-condition, common behavior, and library *functions* in the ATS.

F.1 Configuration functions

```

module Configurations_15118_2 {

  import from Services_HAL_61851 all;
  import from DataStructure_HAL_61851 all;
  import from DataStructure_V2G_TCP_TLS_Port_Control all;
  import from urn_iso_15118_2_2010_AppProtocol language "XSD" all;
  import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
  import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
  import from XSDAUX all;
  import from ComponentsAndPorts all;
  import from LibFunctions_15118_2 all;
  import from Templates_CMN_V2GTcpTlsPortControl all;
  import from Pics_15118 all;

  // ..... Config Functions .....
  function f_SECC_CMN_PR_InitConfiguration_001(out HAL_61851_Listener v_HAL_61851_Listener,
                                              SystemSECC systemSECC) runs on SECC_Tester {

    if(PICS_CMN_CMN_CombinedTesting) {
      map(mtc:pt_SLAC_Port, systemSECC:pt_SLAC_Port);
    }
    map(mtc:pt_V2G_UDP_SDP_Port, systemSECC:pt_V2G_UDP_SDP_Port);

    v_HAL_61851_Listener := HAL_61851_Listener.create("IEC 61851 Listener") alive;
    map(mtc:pt_HAL_61851_Port, systemSECC:pt_HAL_61851_Port);
    map(v_HAL_61851_Listener:pt_HAL_61851_Listener_Port,
        systemSECC:pt_HAL_61851_Listener_Port);
    connect(mtc:pt_HAL_61851_Internal_Port,
            v_HAL_61851_Listener:pt_HAL_61851_Internal_Port);

    v_HAL_61851_Listener.start(f_SECC_HAL61851Listener(false));

    vc_Default_IEC_61851_ListenerBehavior :=
      activate(a_CMN_IEC61851ListenerBehavior(pt_HAL_61851_Internal_Port));
  }

  // ..... Config Functions .....
  function f_SECC_CMN_PR_InitConfiguration_002(SystemSECC systemSECC,
                                              verdicttype v_verdict1,
                                              verdicttype v_verdict2)
    runs on SECC_Tester {

    map(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, systemSECC:pt_V2G_TCP_TLS_ALM_SECC_Port)
    param (vc_v2g_tcp_tls_parameter);

    var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_responseMessage;
    var V2G_TCP_TLS_Port_Control_Ack_TYPE ackType;

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_connectPort)) to "local";

    alt{
      []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_CMN_tcpTlsStatusRes_001(
        e_connectPort,?,?))
        -> value v_responseMessage {

        var boolean isClosed := v_responseMessage.result.portStatus.closed;
        ackType := v_responseMessage.ack;
      }
    }

    if(ackType == e_success) {
      setverdict(v_verdict1, "TCP or TLS handshake was successful.");
    } else if(ackType == e_failed) {

```

```

        setverdict(v_verdict2, "TCP or TLS handshake was not successful.");
    }
}

function f_SECC_CMN_PO_ShutdownConfiguration_001(inout HAL_61851_Listener v_HAL_61851_Listener,
        SystemSECC v_SystemSECC) runs on SECC_Tester {

    unmap(mtc:pt_HAL_61851_Port, v_SystemSECC:pt_HAL_61851_Port);
    unmap(v_HAL_61851_Listener:pt_HAL_61851_Listener_Port,
        v_SystemSECC:pt_HAL_61851_Listener_Port);
    disconnect(mtc:pt_HAL_61851_Internal_Port,
        v_HAL_61851_Listener:pt_HAL_61851_Internal_Port);

    v_HAL_61851_Listener.kill;

    unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, v_SystemSECC:pt_V2G_TCP_TLS_ALM_SECC_Port);
    unmap(mtc:pt_V2G_UDP_SDP_Port, v_SystemSECC:pt_V2G_UDP_SDP_Port);
    if(PICS_CMN_CMN_CombinedTesting) {
        unmap(mtc:pt_SLAC_Port, v_SystemSECC:pt_SLAC_Port);
    }
}

function f_SECC_PO_ShutdownConfiguration_002(SystemSECC v_SystemSECC) runs on SECC_Tester {

    unmap(mtc:pt_SLAC_Port, v_SystemSECC:pt_SLAC_Port);
}

// ..... Config Functions .....
function f_EVCC_CMN_PR_InitConfiguration_001(out HAL_61851_Listener v_HAL_61851_Listener,
        SystemEVCC systemEVCC) runs on EVCC_Tester {
    if(PICS_CMN_CMN_CombinedTesting) {
        map(mtc:pt_SLAC_Port, systemEVCC:pt_SLAC_Port);
    }
    map(mtc:pt_V2G_UDP_SDP_Port, systemEVCC:pt_V2G_UDP_SDP_Port);

    v_HAL_61851_Listener := HAL_61851_Listener.create("IEC 61851 Listener") alive;

    map(mtc:pt_HAL_61851_Port, systemEVCC:pt_HAL_61851_Port);
    map(v_HAL_61851_Listener:pt_HAL_61851_Listener_Port,
        systemEVCC:pt_HAL_61851_Listener_Port);
    connect(mtc:pt_HAL_61851_Internal_Port,
        v_HAL_61851_Listener:pt_HAL_61851_Internal_Port);

    v_HAL_61851_Listener.start(f_EVCC_HAL61851Listener(false));

    vc_Default_IEC_61851_ListenerBehavior :=
        activate(a_CMN_IEC61851ListenerBehavior(pt_HAL_61851_Internal_Port));
}

// ..... Config Functions .....
function f_EVCC_CMN_PR_InitConfiguration_002(SystemEVCC systemEVCC,
        in verdicttype v_vct)
    runs on EVCC_Tester {

    map(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, systemEVCC:pt_V2G_TCP_TLS_ALM_EVCC_Port)
    param (vc_v2g_tcp_tls_parameter);

    var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_responseMessage;
    var V2G_TCP_TLS_Port_Control_Ack_TYPE ackType;
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_connectPort)) to "local";
    alt{
        []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_CMN_tcpTlsStatusRes_001(
            e_connectPort,?,?))
            -> value v_responseMessage {

            var boolean isClosed := v_responseMessage.result.portStatus.closed;
            ackType := v_responseMessage.ack;
        }
    }
    if(ackType == e_success) {
        setverdict(pass, "TCP or TLS setup was successful.");
    } else if(ackType == e_failed) {
        setverdict(v_vct, "TCP or TLS setup was not successful.");
    }
}

pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(

```

```

        e_getPort)) to "local";
    alt{
        [!pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_CMN_tcpTlsStatusRes_002(
            e_getPort,e_success,
            (49152 .. 65535)))
            -> value v_responseMessage {

                vc_V2G_Port_PortNumber := v_responseMessage.result.portNumber.intValue;
            }
    }
}

function f_EVCC_CMN_PO_ShutdownConfiguration_001(inout HAL_61851_Listener v_HAL_61851_Listener,
        SystemEVCC v_SystemEVCC) runs on EVCC_Tester {

    unmap(mtc:pt_HAL_61851_Port, v_SystemEVCC:pt_HAL_61851_Port);
    unmap(v_HAL_61851_Listener:pt_HAL_61851_Listener_Port,
        v_SystemEVCC:pt_HAL_61851_Listener_Port);
    disconnect(mtc:pt_HAL_61851_Internal_Port,
        v_HAL_61851_Listener:pt_HAL_61851_Internal_Port);

    v_HAL_61851_Listener.kill;

    unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, v_SystemEVCC:pt_V2G_TCP_TLS_ALM_EVCC_Port);
    unmap(mtc:pt_V2G_UDP_SDP_Port, v_SystemEVCC:pt_V2G_UDP_SDP_Port);
    if(PICS_CMN_CMN_CombinedTesting) {
        unmap(mtc:pt_SLAC_Port, v_SystemEVCC:pt_SLAC_Port);
    }
}

function f_EVCC_PO_ShutdownConfiguration_002(SystemEVCC v_SystemEVCC) runs on EVCC_Tester {

    unmap(mtc:pt_SLAC_Port, v_SystemEVCC:pt_SLAC_Port);
}
}

```

F.2 Pre-condition functions

F.2.1 SECC functions

```

module PreConditions_SECC_15118_2 {
    import from Services_HAL_61851 all;
    import from PICS_15118_2 all;
    import from Pixit_15118_2 all;
    import from PICS_15118 all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from DataStructure_SDP all;
    import from ComponentsAndPorts all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_SDP all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from LibFunctions_15118_2 all;
    import from TestBehavior_SECC_SupportedAppProtocol all;
    import from TestBehavior_SECC_SessionSetup all;
    import from TestBehavior_SECC_ServiceDiscovery all;
    import from TestBehavior_SECC_Authorization all;
    import from TestBehavior_SECC_PowerDelivery all;
    import from TestBehavior_SECC_PaymentServiceSelection all;
    import from TestBehavior_SECC_ServiceDetail all;
    import from TestBehavior_SECC_PaymentDetails all;
    import from TestBehavior_SECC_CertificateInstallation all;
    import from TestBehavior_SECC_CertificateUpdate all;
    import from TestBehavior_SECC_ChargeParameterDiscovery all;
    import from TestBehavior_SECC_ChargingStatus all;
    import from TestBehavior_SECC_MeteringReceipt all;
    import from TestBehavior_SECC_CableCheck all;
    import from TestBehavior_SECC_PreCharge all;
    import from TestBehavior_SECC_CurrentDemand all;
    import from TestBehavior_SECC_WeldingDetection all;
    import from TestBehavior_SECC_SessionStop all;
    import from Configurations_15118_2 all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Services_Security all;
}

```

```

import from Services_PLCLinkStatus all;
import from Pixit_15118 all;
import from Timer_15118 all;
import from Preconditions_SECC_15118_3 { function f_SECC_CMN_PR_PLCLinkStatus_001};

function f_SECC_CMN_PR_SetPowerFlowConfiguration_001() runs on SECC_Tester return verdicttype {

    var verdicttype verdict := pass;

    return verdict;
}

function f_SECC_CMN_PR_PLCLinkStatusCombinedTesting_001(HAL_61851_Listener
                                                         v_HAL_61851_Listener)
                                                         runs on SECC_Tester
                                                         return verdicttype {

    var verdicttype verdict := f_SECC_CMN_PR_SetPowerFlowConfiguration_001();

    if(PICS_CMN_CMN_CombinedTesting) {
        verdict := f_SECC_CMN_PR_PLCLinkStatus_001(v_HAL_61851_Listener);
    }
    return verdict;
}

function f_SECC_CMN_PR_StateA_001(HAL_61851_Listener v_HAL_61851_Listener)
                                   runs on SECC_Tester return verdicttype {

    var verdicttype verdict := f_SECC_CMN_PR_PLCLinkStatusCombinedTesting_001(
                                                                    v_HAL_61851_Listener);

    if(not PICS_CMN_CMN_CombinedTesting){

        f_SECC_changeValidStateCondition(A);
        verdict := f_SECC_setState(A,v_HAL_61851_Listener);
        pt_HAL_61851_Port.clear;
        sleep((par_CMN_waitForNextHAL));
        verdict := f_SECC_setProximity(0);
    }
    return verdict;
}

function f_SECC_CMN_PR_SetProximityPilot_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                              runs on SECC_Tester return verdicttype {

    var verdicttype verdict := f_SECC_CMN_PR_StateA_001(v_HAL_61851_Listener);

    if(not PICS_CMN_CMN_CombinedTesting){

        v_HAL_61851_Listener.stop;
        v_HAL_61851_Listener.start(f_SECC_HAL61851Listener(true));

        if(PICS_CMN_CMN_ChargingMode == aC){
            f_SECC_changeValidFrequencyRange(0,0);
            f_SECC_changeValidDutyCycleRange(100,100);
            vc_validDutyCycleLowerBound2 := 100;
            vc_validDutyCycleUpperBound2 := 100;
        }
        if ( verdict == pass ) {
            if(PICS_CMN_CMN_PlugType == type1) {
                verdict := f_SECC_setProximity(cc_proximity_type1);
            } else {
                if(PICS_CMN_CMN_ChargingMode == aC){
                    var integer v_proximity_type2_AC;
                    if(PICS_CMN_AC_CableCapability == capability13A) {
                        v_proximity_type2_AC := cc_proximity_type2_AC_13A;
                    } else if(PICS_CMN_AC_CableCapability == capability20A) {
                        v_proximity_type2_AC := cc_proximity_type2_AC_20A;
                    } else if(PICS_CMN_AC_CableCapability == capability32A) {
                        v_proximity_type2_AC := cc_proximity_type2_AC_32A;
                    } else {
                        v_proximity_type2_AC := cc_proximity_type2_AC_63A;
                    }
                    verdict := f_SECC_setProximity(v_proximity_type2_AC);
                } else {
                    verdict := f_SECC_setProximity(cc_proximity_type2_DC);
                }
            }
        }
    }
}

```

```

    }
}
return verdict;
}

function f_SECC_CMN_PR_StateB_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict := f_SECC_CMN_PR_SetProximityPilot_001(v_HAL_61851_Listener);

    if(not PICS_CMN_CMN_CombinedTesting){

        if ( verdict == pass ) {
            sleep((par_CMN_waitForNextHAL));
            f_SECC_changeValidStateCondition(valid_Matching);
            verdict := f_SECC_setState(B,v_HAL_61851_Listener);
        }
        // start CommunicationSetup_Timer
        tc_V2G_EVCC_CommunicationSetup_Timer.start;
        tc_V2G_SECC_CommunicationSetup_Timer.start;
        return verdict;
    }

    function f_SECC_CMN_PR_Binding_001(out HAL_61851_Listener v_HAL_61851_Listener)
        runs on SECC_Tester return verdicttype {

        var verdicttype verdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        if(PICS_CMN_CMN_CombinedTesting) {
            f_SECC_PO_ShutdownConfiguration_002(system);
        }

        // SECC Binding Behavior
        if ( verdict == pass ) {
            var Security_TYPE v_security := cc_hexTls;
            if(PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_SECC_CMN_TLS == false) {
                v_security := cc_hexTcp;
            }
            verdict := f_SECC_CMN_TB_VTB_SDP_001(v_security, inconc);
            if(vc_Security == cc_hexTcp) {
                vc_v2g_tcp_tls_parameter := md_CMN_CMN_TcpParameter_001(vc_V2G_Port_IpAddress,
                                                                    vc_V2G_Port_PortNumber,
                                                                    omit);
            } else if(vc_Security == cc_hexTls) {
                vc_v2gRootCert :=
                fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_V2G_ROOT_VALID");
                if(getverdict == pass) {
                    vc_v2g_tcp_tls_parameter := md_CMN_CMN_TlsParameter_002(
                                                                    vc_V2G_Port_IpAddress,
                                                                    vc_V2G_Port_PortNumber,
                                                                    omit, vc_cipherSuite,
                                                                    omit, vc_v2gRootCert);
                }
            } else {log("Invalid security type was chosen.");}
        }
        return verdict;
    }

    function f_SECC_CMN_PR_SupportedAppProtocol_001(out HAL_61851_Listener
                                                    v_HAL_61851_Listener)
        runs on SECC_Tester return verdicttype {

        var verdicttype verdict := f_SECC_CMN_PR_Binding_001(v_HAL_61851_Listener);
        // SECC_SupportedAppProtocol Behavior
        if ( verdict == pass ) {
            f_SECC_CMN_PR_InitConfiguration_002(system, pass, inconc);
            verdict := f_SECC_CMN_TB_VTB_SupportedAppProtocol_001(inconc);
            //clear SDP Port
            pt_V2G_UDP_SDP_Port.clear;
            vc_Default_SDP_Message := activate(a_SDP_Message());
        }
        return verdict;
    }

    function f_SECC_CMN_PR_SessionSetup_001(out HAL_61851_Listener v_HAL_61851_Listener)
        runs on SECC_Tester return verdicttype {

```

```

var verdicttype verdict :=
f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);
// SECC_SessionSetup Behavior
if ( verdict == pass ) {
    verdict := f_SECC_CMN_TB_VTB_SessionSetup_001(inconc);
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    tc_V2G_SECC_CommunicationSetup_Timer.stop;
}
return verdict;
}

function f_SECC_CMN_PR_ServiceDiscovery_001(out HAL_61851_Listener
v_HAL_61851_Listener)
runs on SECC_Tester return verdicttype {

    var verdicttype verdict := f_SECC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);
    // SECC_ServiceDiscovery Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_CMN_TB_VTB_ServiceDiscovery_001(inconc);
    }
    return verdict;
}

function f_SECC_CMN_PR_ServiceDetail_001(out HAL_61851_Listener v_HAL_61851_Listener)
runs on SECC_Tester return verdicttype {

    var verdicttype verdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);
    // SECC_ServiceDetail Behavior
    if ( verdict == pass and (ispresent(vc_serviceList.service_list) and PIXIT_SECC_CMN_VAS ==
serviceDetail)) {
        if(not(PICS_CMN_CMN_IdentificationMode == eIM and
PIXIT_SECC_CMN_TLS == false)) {
            verdict := f_SECC_CMN_TB_VTB_ServiceDetail_001(inconc);
        }
    }
    return verdict;
}

function f_SECC_CMN_PR_PaymentServiceSelection_001(out HAL_61851_Listener
v_HAL_61851_Listener)
runs on SECC_Tester
return verdicttype {

    var verdicttype verdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);
    // SECC_PaymentServiceSelection Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_CMN_TB_VTB_PaymentServiceSelection_001(inconc);
    }
    return verdict;
}

function f_SECC_CMN_PR_CertificateInstallation_001(out HAL_61851_Listener
v_HAL_61851_Listener)
runs on SECC_Tester
return verdicttype {

    var verdicttype verdict :=
f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);
    // SECC_CertificateInstallation Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_CMN_TB_VTB_CertificateInstallation_001(inconc);
    }
    return verdict;
}

function f_SECC_CMN_PR_CertificateUpdate_001(out HAL_61851_Listener
v_HAL_61851_Listener)
runs on SECC_Tester
return verdicttype {

    var verdicttype verdict :=
f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);
    // SECC_CertificateUpdate Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_CMN_TB_VTB_CertificateUpdate_001(inconc);
    }
    return verdict;
}

```

```

function f_SECC_CMN_PR_PaymentDetails_001(out HAL_61851_Listener
                                         v_HAL_61851_Listener)
    runs on SECC_Tester
    return verdicttype {

    var verdicttype verdict;

    if(PIXIT_SECC_CMN_CertificateHandling == certUpdate and
       PICS_CMN_CMN_IdentificationMode == pnC) {
        verdict := f_SECC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);
    }
    if(PIXIT_SECC_CMN_CertificateHandling == certInstall and
       PICS_CMN_CMN_IdentificationMode == pnC){
        verdict := f_SECC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);
    }
    if((PIXIT_SECC_CMN_CertificateHandling == none_) and
       PICS_CMN_CMN_IdentificationMode == pnC){
        verdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);
    }
    // SECC_PaymentDetails Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_CMN_TB_VTB_PaymentDetails_001(inconc);
    }
    return verdict;
}

function f_SECC_CMN_PR_Authorization_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict;

    if(PICS_CMN_CMN_IdentificationMode == pnC) {
        verdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);
    }
    if(PICS_CMN_CMN_IdentificationMode == eIM){
        verdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);
    }
    // SECC_Authorization Behavior
    if ( verdict == pass ) {
        if(PICS_CMN_CMN_IdentificationMode == eIM){
            verdict := f_SECC_CMN_TB_VTB_Authorization_001(inconc);
        }
        if(PICS_CMN_CMN_IdentificationMode == pnC){
            verdict := f_SECC_CMN_TB_VTB_Authorization_002(inconc);
        }
    }
    return verdict;
}

// PreConditions for AC-TestCases

function f_SECC_AC_PR_ChargeParameterDiscovery_001(out HAL_61851_Listener
                                                    v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);
    // SECC_ChargeParameterDiscovery Behavior
    if ( verdict == pass ) {
        if(PIXIT_SECC_CMN_SalesTariff == unknown) {
            verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(inconc);
        } else {verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
    }
    return verdict;
}

function f_SECC_AC_PR_PowerDeliveryStart_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    // SECC_PowerDeliveryStart Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_AC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener,
                                                       inconc);
    }
    return verdict;
}

```



```

function f_SECC_AC_PR_ChargingStatus_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    // SECC_ChargingStatus Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_AC_TB_VTB_ChargingStatus_001(inconc);
    }
    return verdict;
}

function f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001(out HAL_61851_Listener
    v_HAL_61851_Listener)
    runs on SECC_Tester
    return verdicttype {

    var verdicttype verdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);
    if ( verdict == pass ) {
        var integer loopCounter := 0;
        var integer renegotiationLoopInd := PICS_CMN_CMN_RenegotiationLoopIndication;
        if(not PICS_CMN_CMN_Renegotiation) {renegotiationLoopInd := -1};
        while (vc_ChargeProgress != stop_){
            // ChargeParameterDiscovery
            if ( verdict == pass ) {
                if (vc_ChargeProgress == start_ and loopCounter == 0) {
                    if(PIXIT_SECC_CMN_SalesTariff == unknown) {
                        verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(inconc);
                    } else {
                        verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
                }
            } else { return verdict; }

            // PowerDelivery
            if ( verdict == pass) {
                if (vc_ChargeProgress == start_){
                    verdict := f_SECC_AC_TB_VTB_PowerDelivery_001(start_,
                                                                v_HAL_61851_Listener,
                                                                inconc);
                }
            } else { return verdict; }

            while(vc_EVSENotification == none_ and loopCounter <
                PICS_CMN_CMN_LoopCounter and loopCounter != renegotiationLoopInd) {
                // ChargingStatus
                if (verdict == pass) {
                    if(true){ verdict := f_SECC_AC_TB_VTB_ChargingStatus_001(inconc); }}
                    else { return verdict; }

                // MeteringReceipt
                if (verdict == pass) {
                    if(vc_receiptRequired and PICS_CMN_CMN_IdentificationMode == pnC){
                        verdict := f_SECC_AC_TB_VTB_MeteringReceipt_001(inconc); }}
                    else { return verdict; }

                loopCounter := loopCounter + 1;
                if (verdict == pass) {
                    log("loopcounter -> ChargingStatus: ",loopCounter);
                    sleep(par_SECC_chargingLoop_pause);
                }
            }

            if (vc_EVSENotification == reNegotiation or
                loopCounter == renegotiationLoopInd and PICS_CMN_CMN_Renegotiation){
                if(verdict == pass) {vc_ChargeProgress := renegotiate; }
                else { return verdict; }
                if(verdict == pass) {
                    verdict := f_SECC_AC_TB_VTB_PowerDelivery_001(renegotiate,
                                                                v_HAL_61851_Listener,
                                                                inconc)
                } else { return verdict; }
                if(verdict == pass) {
                    vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 10 };
                    if(PIXIT_SECC_CMN_SalesTariff == unknown) {
                        verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(inconc);
                    } else {
                        verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
                }
                if(verdict == pass) {

```

```

        vc_ChargeProgress := start_; renegotiationLoopInd := 0;
        vc_EVSENotification := none_
    } else { return verdict; }
}
if(loopCounter == PICS_CMN_CMN_LoopCounter or
   vc_EVSENotification == stopCharging){vc_ChargeProgress := stop_;}
}
return verdict;
}

function f_SECC_AC_PR_ChargingStatusOrMeteringReceiptRenegotiation_001(out HAL_61851_Listener
                                                                    v_HAL_61851_Listener)
                                                                    runs on SECC_Tester
                                                                    return verdicttype {

var verdicttype verdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);
if ( verdict == pass ) {

    while (vc_ChargeProgress != renegotiate){
        // ChargeParameterDiscovery
        if (verdict == pass and vc_ChargeProgress == start_) {
            if(PIXIT_SECC_CMN_SalesTariff == unknown) {
                verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(inconc);
            } else {verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
        } else { return verdict; }

        // PowerDelivery
        if (verdict == pass and vc_ChargeProgress == start_) {
            verdict := f_SECC_AC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener,
                                                         inconc);
        } else { return verdict; }

        var integer loopCounter := 0;
        while(vc_EVSENotification == none_ and
              loopCounter < PICS_CMN_CMN_LoopCounter and
              loopCounter != PICS_CMN_CMN_RenegotiationLoopIndication) {
            // ChargingStatus
            if (verdict == pass) {
                if(true){ verdict := f_SECC_AC_TB_VTB_ChargingStatus_001(inconc); }}
            else { return verdict; }

            // MeteringReceipt
            if (verdict == pass) {
                if(vc_receiptRequired and
                   PICS_CMN_CMN_IdentificationMode == pnC){
                    verdict := f_SECC_AC_TB_VTB_MeteringReceipt_001(inconc); }}
            else { return verdict; }
            loopCounter := loopCounter + 1;
            if (verdict == pass) {
                log("loopcounter -> ChargingStatus: ",loopCounter);
                sleep(par_SECC_chargingLoop_pause);
            }
        }

        if (vc_EVSENotification == reNegotiation or
            loopCounter == PICS_CMN_CMN_RenegotiationLoopIndication and
            PICS_CMN_CMN_Renegotiation){
            if(verdict == pass ) {
                vc_ChargeProgress := renegotiate;}
            else { return verdict; }
        }
    }
}
return verdict;
}

function f_SECC_AC_PR_PowerDeliveryRenegotiation_001(out HAL_61851_Listener
                                                                    v_HAL_61851_Listener)
                                                                    runs on SECC_Tester return verdicttype {

var verdicttype verdict;
verdict :=
f_SECC_AC_PR_ChargingStatusOrMeteringReceiptRenegotiation_001(v_HAL_61851_Listener);

// f_SECC_PowerDeliveryStopAC Behavior
if ( verdict == pass ) {
    verdict := f_SECC_AC_TB_VTB_PowerDelivery_001(renegotiate, v_HAL_61851_Listener,
                                                         inconc);
}
}

```

```

    }
    return verdict;
}

function f_SECC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(out HAL_61851_Listener
                                                                v_HAL_61851_Listener)
    runs on SECC_Tester
    return verdicttype {

    var verdicttype verdict;
    verdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    // f_SECC_ChargeParameterDiscovery Behavior
    if ( verdict == pass ) {
        vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 10 };
        if(PIXIT_SECC_CMN_SalesTariff == unknown) {
            verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(inconc);
        } else {verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
    }
    return verdict;
}

function f_SECC_AC_PR_PowerDeliveryStop_001(out HAL_61851_Listener
                                             v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict;

    verdict := f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001(v_HAL_61851_Listener);

    // SECC_f_SECC_PowerDeliveryStopAC Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_AC_TB_VTB_PowerDelivery_001(stop, v_HAL_61851_Listener,
                                                       inconc);
    }
    return verdict;
}

function f_SECC_AC_PR_SessionStop_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict;

    verdict := f_SECC_AC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener);

    // SECC_WeldingDetection Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_CMN_TB_VTB_SessionStop_001(pause, v_HAL_61851_Listener,
                                                       inconc);
    }
    return verdict;
}

function f_SECC_AC_PR_SessionStop_002(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict;

    verdict := f_SECC_AC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener);

    // SECC_WeldingDetection Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener,
                                                       inconc);
    }
    return verdict;
}

// PreConditions for DC-TestCases

function f_SECC_DC_PR_ChargeParameterDiscovery_001(out HAL_61851_Listener
                                                    v_HAL_61851_Listener)
    runs on SECC_Tester
    return verdicttype {

    var verdicttype verdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    // SECC_ChargeParameterDiscovery Behavior
    if ( verdict == pass ) {

```

```

        if(PIXIT_SECC_CMN_SalesTariff == unknown) {
            verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(inconc);
        } else {verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
    }
    return verdict;
}

function f_SECC_DC_PR_CableCheck_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict :=
        f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    // SECC_CableCheck Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, inconc);
    }
    if(PIXIT_CMN_DC_PowerFlow and verdict == pass){
        if(PICS_CMN_DC_TwoQuadrantOperation) {
            ExternalFunctions_PSI9000.fx_setDCPowerSwitchStatus(true);
        }
        fx_setDCPowerSwitchStatus(true);
    }
    return verdict;
}

function f_SECC_DC_PR_PreCharge_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict := f_SECC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

    // SECC_PreCharge Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_DC_TB_VTB_PreCharge_001(inconc);
    }
    return verdict;
}

function f_SECC_DC_PR_PowerDeliveryStart_001(out HAL_61851_Listener
    v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    // SECC_PowerDeliveryStart Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_DC_TB_VTB_PowerDelivery_001(start, v_HAL_61851_Listener,
            false, true, inconc);
    }
    return verdict;
}

function f_SECC_DC_PR_CurrentDemand_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict :=
        f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    if ( verdict == pass ) {
        // SECC_PowerDeliveryStart Behavior
        verdict := f_SECC_DC_TB_VTB_CurrentDemand_001(inconc);
    }
    return verdict;
}

function f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(out HAL_61851_Listener
    v_HAL_61851_Listener)
    runs on SECC_Tester
    return verdicttype {

    var verdicttype verdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);
    if ( verdict == pass ) {
        var integer loopCounter := 0;
        var integer renegotiationLoopInd := PICS_CMN_CMN_RenegotiationLoopIndication;
        if(not PICS_CMN_CMN_Renegotiation) {renegotiationLoopInd := -1};
        while (vc_ChargeProgress != stop_){

            // ChargeParameterDiscovery
            if ( verdict == pass ) {

```

```

        if (vc_ChargeProgress == start_ and loopCounter == 0) {
            if (PIXIT_SECC_CMN_SalesTariff == unknown) {
                verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(inconc);
            } else { verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
        }
    } else { return verdict; }

    // CableCheck
    if ( verdict == pass) {
        if (vc_ChargeProgress == start_) {
            verdict := f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, inconc);
        }
    } else { return verdict; }

    // PreCharge
    if ( verdict == pass) {
        if (vc_ChargeProgress == start_) {
            verdict := f_SECC_DC_TB_VTB_PreCharge_001(inconc);
        }
    } else { return verdict; }

    // PowerDelivery
    if ( verdict == pass) {
        if (vc_ChargeProgress == start_) {
            verdict := f_SECC_DC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener,
                false, true, inconc);
        }
    } else { return verdict; }

    while (vc_EVSENotification == none_ and
        loopCounter < PICS_CMN_CMN_LoopCounter and
        loopCounter != renegotiationLoopInd) {
        // CurrentDemand
        if ( verdict == pass) {
            if (true) { verdict := f_SECC_DC_TB_VTB_CurrentDemand_001(inconc); }
        } else { return verdict; }

        // MeteringReceipt
        if ( verdict == pass) {
            if (vc_receiptRequired and PICS_CMN_CMN_IdentificationMode == pnC) {
                verdict := f_SECC_DC_TB_VTB_MeteringReceipt_001(inconc);
            } else { return verdict; }
            loopCounter := loopCounter + 1;
            if ( verdict == pass) {
                log("loopcounter -> CurrentDemand: ", loopCounter);
                sleep(par_SECC_chargingLoop_pause);
            }
        }
    }
    if (vc_EVSENotification == reNegotiation or
        loopCounter == renegotiationLoopInd and PICS_CMN_CMN_Renegotiation) {
        if ( verdict == pass) { vc_ChargeProgress := renegotiate; } else { return verdict; }
        if ( verdict == pass) {
            verdict := f_SECC_DC_TB_VTB_PowerDelivery_001(renegotiate,
                v_HAL_61851_Listener,
                false, true, inconc);
        } else { return verdict; }
        if ( verdict == pass) {
            vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 50 };
            if (PIXIT_SECC_CMN_SalesTariff == unknown) {
                verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(inconc);
            } else { verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
            if ( verdict == pass) {
                vc_ChargeProgress := start_; renegotiationLoopInd := 0;
                vc_EVSENotification := none_
            } else { return verdict; }
        }
    }
    if (loopCounter == PICS_CMN_CMN_LoopCounter or
        vc_EVSENotification == stopCharging) { vc_ChargeProgress := stop; }
}
return verdict;
}

```

```

, PICS_CMN_CMN_GracefulHandling, PICS_CMN_CMN_GracefulHandling, PICS_CMN_CMN_GracefulHandling,
PICS_CMN_CMN_GracefulHandling, PICS_CMN_CMN_GracefulHandling, PICS_CMN_CMN_GracefulHandling,
PICS_CMN_CMN_GracefulHandling, PICS_CMN_CMN_GracefulHandling, PICS_CMN_CMN_GracefulHandling,
PICS_CMN_CMN_GracefulHandling

```

```
function f_SECC_DC_PR_CurrentDemandOrMeteringReceiptRenegotiation_001(out HAL_61851_Listener
```

```

v_HAL_61851_Listener)
runs on SECC_Tester
return verdicttype {

var verdicttype verdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);
if ( verdict == pass ) {

    while (vc_ChargeProgress != renegotiate){

        // ChargeParameterDiscovery
        if ( verdict == pass and vc_ChargeProgress == start_ ) {
            if(PIXIT_SECC_CMN_SalesTariff == unknown) {
                verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(inconc);
            } else {verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
        } else { return verdict; }

        // CableCheck
        if ( verdict == pass and vc_ChargeProgress == start_ ) {
            verdict := f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener,inconc);
        } else { return verdict; }

        // PreCharge
        if ( verdict == pass and vc_ChargeProgress == start_ ) {
            verdict := f_SECC_DC_TB_VTB_PreCharge_001(inconc);
        } else { return verdict; }

        // PowerDelivery
        if ( verdict == pass and vc_ChargeProgress == start_ ) {
            verdict := f_SECC_DC_TB_VTB_PowerDelivery_001(start_,
                v_HAL_61851_Listener,
                false, true, inconc);
        } else { return verdict; }

        var integer loopCounter := 0;
        while(vc_EVSENotification == none_ and
            loopCounter < PICS_CMN_CMN_LoopCounter and
            loopCounter != PICS_CMN_CMN_RenegotiationLoopIndication){
            // CurrentDemand
            if (verdict == pass) {
                if(true){ verdict := f_SECC_DC_TB_VTB_CurrentDemand_001(inconc); }}
            else { return verdict; }

            // MeteringReceipt
            if (verdict == pass) {
                if(vc_receiptRequired and PICS_CMN_CMN_IdentificationMode == pnC){
                    verdict := f_SECC_DC_TB_VTB_MeteringReceipt_001(inconc); }
            } else { return verdict; }
            loopCounter := loopCounter + 1;
            if (verdict == pass) {
                log("loopcounter -> CurrentDemand: ",loopCounter);
                Sleep(par_SECC_chargingLoop_pause);
            }
        }
        if (vc_EVSENotification == reNegotiation or
            loopCounter == PICS_CMN_CMN_RenegotiationLoopIndication and
            PICS_CMN_CMN_Renegotiation){
            if(verdict == pass ) {vc_ChargeProgress := renegotiate;}
            else { return verdict; }
        }
    }
    return verdict;
}

function f_SECC_DC_PR_CurrentDemandOrMeteringReceiptRenegotiation_002(out HAL_61851_Listener
    v_HAL_61851_Listener)
runs on SECC_Tester
return verdicttype {

var verdicttype verdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

if ( verdict == pass ) {
    var integer moduloValueSOC;

    if(PICS_CMN_CMN_LoopCounter >= 100) {
        //soc init values
        var integer socDifference := 100 - par_EVRESSSOC;
        var float socValue := ( int2float(socDifference) /
            int2float(PICS_CMN_CMN_LoopCounter));
    }
}

```

```

    moduloValueSOC := float2int( 1.0 / socValue);
}

while (vc_ChargeProgress != renegotiate){

    // ChargeParameterDiscovery
    if ( verdict == pass and vc_ChargeProgress == start_) {
        if(PIXIT_SECC_CMN_SalesTariff == unknown) {
            verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(inconc);
        } else {verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
    } else { return verdict; }

    // CableCheck
    if ( verdict == pass and vc_ChargeProgress == start_) {
        verdict := f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, inconc);
    } else { return verdict; }

    // PreCharge
    if ( verdict == pass and vc_ChargeProgress == start_) {
        verdict := f_SECC_DC_TB_VTB_PreCharge_001(inconc);
    } else { return verdict; }

    // PowerDelivery
    if ( verdict == pass and vc_ChargeProgress == start_) {
        verdict := f_SECC_DC_TB_VTB_PowerDelivery_001(start_,
                                                       v_HAL_61851_Listener,
                                                       false, true, inconc);
    } else { return verdict; }

    var integer loopCounter := 0;

    while(vc_EVSENotification == none_ and
          loopCounter < PICS_CMN_CMN_LoopCounter and
          loopCounter != PICS_CMN_CMN_RenegotiationLoopIndication){

        if(PICS_CMN_CMN_LoopCounter >= 100) {

            var float measVoltage := fx_getMeasuredVoltage();
            log("ELR9000 measured voltage ", measVoltage);
            var float measCurrent := fx_getMeasuredCurrent();
            log ("ELR9000 measured current ", measCurrent);

            if (loopCounter == 0) {
                fx_setCurrent(par_LoadStartValCurrent);
                var float getCurrent := fx_getCurrent();
                log("INITIAL Set current: ", getCurrent);

                vc_EVTARGETCURRENT.value_ := vc_EVTARGETCURRENT.value_+1;
            }

            var integer loopDifference := PICS_CMN_CMN_LoopCounter - loopCounter;
            // increase voltage loop
            if (loopCounter > 10 and measVoltage >=
                f_CMN_PhysicalValue_GetValue(par_EVBatteryVoltage)) {

                var float voltageDifference := int2float(vc_EVTARGETVOLTAGE.value_) -
measVoltage;

                var float voltageStep := voltageDifference / int2float(loopDifference);
                var float voltageIncrease :=measVoltage+voltageStep;

                if (voltageIncrease <= int2float(vc_EVTARGETVOLTAGE.value_)) {
                    fx_setVoltage(voltageIncrease);
                }
            }

            //soc loop
            if (loopCounter mod moduloValueSOC == 0) {
                // increase SOC
                if (vc_DC_EVStatus.eVRESSSOC+1 <=par_EVBatteryStepSize) {
                    vc_DC_EVStatus.eVRESSSOC := vc_DC_EVStatus.eVRESSSOC+1;
                    log("vc_DC_EVStatus", vc_DC_EVStatus);
                }
            }

            // increase current loop
            if (loopCounter > 1 and loopCounter mod 100 == 0) {
                var float currentValue := fx_getCurrent();

```



```

        if (currentValue < f_CMN_PhysicalValue_GetValue(
            par_EVMaximumCurrentLimit)) {
            fx_setCurrent(currentValue+1.0);
            vc_EVTargetCurrent.value_ := vc_EVTargetCurrent.value_+1;
        }
    }

} else {
    fx_setCurrent(par_LoadStartValCurrent);
    var float getCurrent := fx_getCurrent();
    log("INITIAL Set current: ", getCurrent);

    vc_EVTargetCurrent.value_ := float2int(par_LoadStartValCurrent);
}
// CurrentDemand
if (verdict == pass) {
    if(true){ verdict := f_SECC_DC_TB_VTB_CurrentDemand_001(inconc); }
else { return verdict; }

// MeteringReceipt
if (verdict == pass) {
    if(vc_receiptRequired and PICS_CMN_CMN_IdentificationMode == pnC){
        verdict := f_SECC_DC_TB_VTB_MeteringReceipt_001(inconc); }
    } else { return verdict; }
    loopCounter := loopCounter + 1;
    if (verdict == pass) {
        log("loopcounter -> CurrentDemand: ",loopCounter);
        sleep(par_SECC_chargingLoop_pause);
    }
}
if (vc_EVSENotification == reNegotiation or
    loopCounter == PICS_CMN_CMN_RenegotiationLoopIndication and
    PICS_CMN_CMN_Renegotiation){
    if(verdict == pass ) {vc_ChargeProgress := renegotiate;}
    else { return verdict; }
}
}
}
return verdict;
}

function f_SECC_DC_PR_PowerDeliveryRenegotiation_001(out HAL_61851_Listener
    v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict;

    verdict := f_SECC_DC_PR_CurrentDemandOrMeteringReceiptRenegotiation_001(
        v_HAL_61851_Listener);

    // f_SECC_PowerDeliveryStopDC Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_DC_TB_VTB_PowerDelivery_001(renegotiate, v_HAL_61851_Listener,
            false, true, inconc);
    }
    return verdict;
}

function f_SECC_DC_PR_ChargeParameterDiscoveryRenegotiation_001(out HAL_61851_Listener
    v_HAL_61851_Listener)
    runs on SECC_Tester
    return verdicttype {

    var verdicttype verdict;
    verdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    // f_SECC_ChargeParameterDiscovery Behavior
    if ( verdict == pass ) {
        vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 50 };
        if(PIXIT_SECC_CMN_SalesTariff == unknown) {
            verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(inconc);
        } else {verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
    }
    // CableCheck
    if ( verdict == pass ) {
        verdict := f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, inconc);
    } else { return verdict; }

    // PreCharge

```

```

    if ( verdict == pass ) {
        verdict := f_SECC_DC_TB_VTB_PreCharge_001(inconc);
    } else { return verdict; }

    return verdict;
}

function f_SECC_DC_PR_PowerDeliveryStop_001(out HAL_61851_Listener v_HAL_61851_Listener,
      in boolean v_setState) runs on SECC_Tester
    return verdicttype {

    var verdicttype verdict;

    verdict := f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(
        v_HAL_61851_Listener);

    // SECC_PowerDeliveryStopDC Behavior
    if ( verdict == pass ) {
        vc_DC_EVStatus.eVReady := false;
        verdict := f_SECC_DC_TB_VTB_PowerDelivery_001(stop, v_HAL_61851_Listener,
            true, v_setState, inconc);
    }
    return verdict;
}

function f_SECC_DC_PR_WeldingDetection_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict;

    verdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

    // SECC_WeldingDetection Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_DC_TB_VTB_WeldingDetection_001(inconc);
    }
    return verdict;
}

function f_SECC_DC_PR_SessionStop_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict;

    if(PICS_SECC_DC_WeldingDetection) {
        verdict := f_SECC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);
    }
    else {
        verdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);
    }

    // SECC_WeldingDetection Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_CMN_TB_VTB_SessionStop_001(pause, v_HAL_61851_Listener,
            inconc);
    }
    return verdict;
}

function f_SECC_DC_PR_SessionStop_002(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var verdicttype verdict;

    if(PICS_SECC_DC_WeldingDetection) {
        verdict := f_SECC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);
    }
    else {
        verdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);
    }

    // SECC_WeldingDetection Behavior
    if ( verdict == pass ) {
        verdict := f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener,
            inconc);
    }
    return verdict;
}

```

```

// Paused Session

function f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(out HAL_61851_Listener
                                                             v_HAL_61851_Listener)
    runs on SECC_Tester
    return verdicttype {

        var verdicttype verdict := pass;

        v_HAL_61851_Listener.stop;
        v_HAL_61851_Listener.start(f_SECC_HAL61851Listener(true));

        if(PICS_CMN_CMN_ChargingMode == aC){
            f_SECC_changeValidFrequencyRange(0,0);
            f_SECC_changeValidDutyCycleRange(100,100);
            vc_validDutyCycleLowerBound2 := 100;
            vc_validDutyCycleUpperBound2 := 100;
        }

        if(PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {

            // B toggle
            tc_TP_EV_vald_state_duration.start(par_TP_EV_vald_state_duration);
            f_SECC_setState(B,v_HAL_61851_Listener);
            tc_TP_EV_vald_state_duration.timeout;
            // C toggle
            tc_TP_EV_vald_state_duration.start(par_TP_EV_vald_state_duration);
            f_SECC_changeValidStateCondition(C);
            f_SECC_setState(C,v_HAL_61851_Listener);
            tc_TP_EV_vald_state_duration.timeout;
            // B toggle
            tc_TP_EV_vald_state_duration.start(par_TP_EV_vald_state_duration);
            f_SECC_changeValidStateCondition(B);
            f_SECC_setState(B,v_HAL_61851_Listener);
            tc_TP_EV_vald_state_duration.timeout;

        } else {
            f_SECC_confirmDutyCycle(v_HAL_61851_Listener,
                                   (PICS_CMN_CMN_WakeUp -
                                    PIXIT_CMN_CMN_WakeUp + 5.0),
                                   inconc);

            if (getverdict != pass) {
                log("The SUT did not initiate a wake-up " &
                    "within 'PICS_CMN_CMN_WakeUp'.");
            }
        }

        tc_T_conn_max_comm.start(par_T_conn_max_comm);

        f_SECC_getPLCLinkEstablishmentAfterSleepMode(pass, inconc, inconc);
        unmap(mtc:pt_SLAC_Port, system:pt_SLAC_Port);

        if(getverdict == pass) {
            f_SECC_setIsConfirmationFlagDC();
            verdict := f_SECC_confirmDutyCycle(v_HAL_61851_Listener,
                                                (par_T_conn_max_comm - tc_T_conn_max_comm.read),
                                                inconc);
        }

        if(getverdict == pass) {
            tc_V2G_EVCC_CommunicationSetup_Timer.start;

            // SECC_Binding_Behavior
            var Security_TYPE v_security := cc_hexTls;
            if(PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_SECC_CMN_TLS == false) {
                v_security := cc_hexTcp;
            }
            verdict := f_SECC_CMN_TB_VTB_SDP_001(v_security, inconc);
            if(vc_Security == cc_hexTcp) {
                vc_v2g_tcp_tls_parameter := md_CMN_CMN_TcpParameter_001(vc_V2G_Port_IpAddress,
                                                                           vc_V2G_Port_PortNumber,
                                                                           omit);
            } else if(vc_Security == cc_hexTls) {
                vc_v2gRootCert :=
                fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_V2G_ROOT_VALID");
                if(getverdict == pass) {
                    vc_v2g_tcp_tls_parameter := md_CMN_CMN_TlsParameter_002(
                                                                vc_V2G_Port_IpAddress,

```

```

        vc_V2G_Port_PortNumber,
        omit, vc_cipherSuite,
        omit, vc_v2gRootCert);
    }
} else {log("Invalid security type was chosen.");}

// SECC_SupportedAppProtocol Behavior
if ( verdict == pass ) {
    f_SECC_CMN_PR_InitConfiguration_002(system, pass, inconc);
    verdict := f_SECC_CMN_TB_VTB_SupportedAppProtocol_001(inconc);
    //clear SDP Port
    pt_V2G_UDP_SDP_Port.clear;
    vc_Default_SDP_Message := activate(a_SDP_Message());
}
}
return getverdict;
}

function f_SECC_CMN_PR_SessionSetupPausedSession_001(out HAL_61851_Listener
v_HAL_61851_Listener)
runs on SECC_Tester
return verdicttype {

var verdicttype verdict;

verdict := f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);

// SECC_SessionSetup Behavior
if ( verdict == pass ) {
    verdict := f_SECC_CMN_TB_VTB_SessionSetup_005(inconc);
}
return verdict;
}

function f_SECC_CMN_PR_AuthorizationPausedSession_001(out HAL_61851_Listener
v_HAL_61851_Listener)
runs on SECC_Tester
return verdicttype {

var verdicttype verdict;

verdict := f_SECC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);

// SECC_ServiceDiscovery Behavior
if ( verdict == pass ) {
    verdict := f_SECC_CMN_TB_VTB_ServiceDiscovery_005(inconc);
}
// SECC_ServiceDetail Behavior
if ( verdict == pass and (ispresent(vc_serviceList.service_list))) {
    verdict := f_SECC_CMN_TB_VTB_ServiceDetail_001(inconc);
}
// SECC_PaymentServiceSelection Behavior
if ( verdict == pass ) {
    verdict := f_SECC_CMN_TB_VTB_PaymentServiceSelection_001(inconc);
}
if ( verdict == pass ) {
    if(PIXIT_SECC_CMN_CertificateHandling == certInstall and
PICS_CMN_CMN_IdentificationMode == pnC) {
        verdict := f_SECC_CMN_TB_VTB_CertificateInstallation_001(inconc);
    }
}
if ( verdict == pass ) {
    if(PIXIT_SECC_CMN_CertificateHandling == certUpdate and
PICS_CMN_CMN_IdentificationMode == pnC) {
        verdict := f_SECC_CMN_TB_VTB_CertificateUpdate_001(inconc);
    }
}
// SECC_PaymentDetails Behavior
if ( verdict == pass ) {
    if(PICS_CMN_CMN_IdentificationMode == pnC) {
        verdict := f_SECC_CMN_TB_VTB_PaymentDetails_001(inconc);
    }
}
// SECC_Authorization Behavior
if ( verdict == pass ) {
    if(PICS_CMN_CMN_IdentificationMode == eIM){
        verdict := f_SECC_CMN_TB_VTB_Authorization_001(inconc);
    }
    if(PICS_CMN_CMN_IdentificationMode == pnC){

```

```

        verdict := f_SECC_CMN_TB_VTB_Authorization_002(inconc);
    }
    }
    return verdict;
}
}

```

F.2.2 EVCC functions

```

module PreConditions_EVCC_15118_2 {

    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Pics_15118_2 all;
    import from Pixit_15118_2 all;
    import from Pics_15118 all;
    import from ComponentsAndPorts all;
    import from DataStructure_HAL_61851 all;
    import from DataStructure_SDP all;
    import from Configurations_15118_2 all;
    import from Timer_15118_2 all;
    import from LibFunctions_15118_2 all;
    import from TestBehavior_EVCC_SDP all;
    import from TestBehavior_EVCC_SupportedAppProtocol all;
    import from TestBehavior_EVCC_SessionSetup all;
    import from TestBehavior_EVCC_ServiceDetail all;
    import from TestBehavior_EVCC_ServiceDiscovery all;
    import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
    import from TestBehavior_EVCC_CertificateHandlingAndPaymentDetails all;
    import from TestBehavior_EVCC_CertificateInstallation all;
    import from TestBehavior_EVCC_CertificateUpdate all;
    import from TestBehavior_EVCC_Authorization all;
    import from TestBehavior_EVCC_ChargeParameterDiscovery all;
    import from TestBehavior_EVCC_PowerDelivery all;
    import from TestBehavior_EVCC_ChargingStatus all;
    import from TestBehavior_EVCC_ChargingLoopAcOrPowerDelivery all;
    import from TestBehavior_EVCC_MeteringReceipt all;
    import from TestBehavior_EVCC_CableCheck all;
    import from TestBehavior_EVCC_PreCharge all;
    import from TestBehavior_EVCC_PreChargeOrPowerDelivery all;
    import from TestBehavior_EVCC_CurrentDemand all;
    import from TestBehavior_EVCC_ChargingLoopDcOrPowerDelivery all;
    import from TestBehavior_EVCC_SessionStop all;
    import from TestBehavior_EVCC_WeldingDetection all;
    import from TestBehavior_EVCC_WeldingDetectionOrSessionStop all;
    import from Services_Security all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Templates_CMN_Security all;
    import from urn_iso_15118_2_2013_X509Def all;
    import from Pixit_15118 all;
    import from Timer_15118 all;
    import from PreConditions_EVCC_15118_3 { function f_EVCC_CMN_PR_PLCLinkStatus_001};
    import from Services_PLCLinkStatus all;
    import from Timer_15118_3 all;

    // PreConditions for EVCC-TestCases

    function f_EVCC_CMN_PR_SetPowerFlowConfiguration_001() runs on EVCC_Tester return verdicttype {

        var verdicttype verdict := pass;

        return verdict;
    }

    function f_EVCC_CMN_PR_GetTLSData_001(out HAL_61851_Listener
                                         v_HAL_61851_Listener)
        runs on EVCC_Tester
        return verdicttype {

            var verdicttype verdict := f_EVCC_CMN_PR_SetPowerFlowConfiguration_001();

            vc_leafCert := fx_loadCertificate(PICS_CMN_CMN_PkiName,
                                             "CRT_EVSE_LEAF_VALID");
            vc_subCert2 := fx_loadCertificate(PICS_CMN_CMN_PkiName,
                                             "CRT_CPO_SUB2_VALID");
            vc_subCert1 := fx_loadCertificate(PICS_CMN_CMN_PkiName,

```

```

        "CRT_CPO_SUB1_VALID");

    return verdict;
}

function f_EVCC_CMN_PR_PLCLinkStatusCombinedTesting_001(out HAL_61851_Listener
                                                         v_HAL_61851_Listener)
    runs on EVCC_Tester
    return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_GetTLSData_001();

    if(PICS_CMN_CMN_CombinedTesting) {
        verdict := f_EVCC_CMN_PR_PLCLinkStatus_001(v_HAL_61851_Listener);
    }
    return verdict;
}

function f_EVCC_CMN_PR_DisableControlPilot_001(out HAL_61851_Listener
                                                 v_HAL_61851_Listener)
    runs on EVCC_Tester
    return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_PLCLinkStatusCombinedTesting_001(
                                                         v_HAL_61851_Listener);

    if(not PICS_CMN_CMN_CombinedTesting){
        verdict := f_EVCC_setPwmMode(e_OscOff);
        sleep((par_CMN_waitForNextHAL));
        verdict := f_EVCC_setProximity(0);
    }
    return verdict;
}

function f_EVCC_CMN_PR_DutyCycle_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_DisableControlPilot_001(
                                                         v_HAL_61851_Listener);

    if(not PICS_CMN_CMN_CombinedTesting){
        if ( verdict == pass ) {
            if(PICS_CMN_CMN_ChargingMode == ac){
                verdict := f_EVCC_setDutyCycle(vc_DutyCycle);
            }
            else {
                verdict := f_EVCC_setDutyCycle(5);
            }
        }
    }
    return verdict;
}

function f_EVCC_CMN_PR_SetProximityPilot_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_DutyCycle_001(v_HAL_61851_Listener);

    if(not PICS_CMN_CMN_CombinedTesting){
        v_HAL_61851_Listener.stop;
        v_HAL_61851_Listener.start(f_EVCC_HAL61851Listener(true));

        f_EVCC_changeValidStateCondition(E,B);

        if ( verdict == pass) {
            if(PICS_CMN_CMN_PlugType == type1) {
                verdict := f_EVCC_setProximity(cc_proximity_type1);
            } else {
                if(PICS_CMN_CMN_ChargingMode == ac){
                    var integer v_proximity_type2_AC;
                    if(PICS_CMN_AC_CableCapability == capability13A) {
                        v_proximity_type2_AC := cc_proximity_type2_AC_13A;
                    } else if(PICS_CMN_AC_CableCapability == capability20A) {
                        v_proximity_type2_AC := cc_proximity_type2_AC_20A;
                    } else if(PICS_CMN_AC_CableCapability == capability32A) {
                        v_proximity_type2_AC := cc_proximity_type2_AC_32A;
                    } else {
                        v_proximity_type2_AC := cc_proximity_type2_AC_63A;
                    }
                }
            }
        }
    }
}

```

```

        verdict := f_EVCC_setProximity(v_proximity_type2_AC);
    } else {
        verdict := f_EVCC_setProximity(cc_proximity_type2_DC);
    }
}
}
}
return verdict;
}

function f_EVCC_CMN_PR_EnableControlPilot_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_SetProximityPilot_001(v_HAL_61851_Listener);

    if(not PICS_CMN_CMN_CombinedTesting){
        if ( verdict == pass ) {
            sleep((par_CMN_waitForNextHAL));
            verdict := f_EVCC_setPwmMode(e_OscOn);
        }
    }
    return verdict;
}

function f_EVCC_CMN_PR_StateB_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_EnableControlPilot_001(v_HAL_61851_Listener);

    if(not PICS_CMN_CMN_CombinedTesting){
        if ( verdict == pass ) {
            timer statetimer := par_CMN_HAL_Timeout;
            verdict := f_EVCC_confirmState(B, v_HAL_61851_Listener,
                statetimer, inconc);
        }
    }
    tc_V2G_SECC_CommunicationSetup_Timer.start;
    return verdict;
}

function f_EVCC_CMN_PR_SDP_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    if(PICS_CMN_CMN_CombinedTesting) {
        f_EVCC_PO_ShutdownConfiguration_002(system);
    }

    // EVCC_Binding_Behavior
    if ( verdict == pass ) {
        var template_Security_TYPE v_security := ?;
        verdict := f_EVCC_CMN_TB_VTB_SDP_001(v_security, inconc);
        if(PICS_CMN_CMN_IdentificationMode == pnC) {
            if(vc_Security != cc_hexTls){
                setverdict(inconc, "SUT did not request the correct transmission security.");
            }
        }
        if(PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_ ) {
            if(vc_Security != cc_hexTcp){
                setverdict(inconc, "SUT did not request the correct transmission security.");
            }
        }
        if(getverdict == pass) {
            if(vc_Security == cc_hexTcp) {
                vc_v2g_tcp_tls_parameter := md_CMN_CMN_TcpParameter_001(
                    vc_sut_ipAddress.ipAddress,
                    vc_V2G_Port_PortNumber,
                    vc_Security);
            } else if(vc_Security == cc_hexTls) {
                vc_v2g_tcp_tls_parameter := md_CMN_CMN_TlsParameter_001(
                    vc_sut_ipAddress.ipAddress,
                    vc_V2G_Port_PortNumber,
                    vc_Security, vc_cipherSuite,
                    vc_enabledProtocol,
                    md_CertificateChainType_002(
                        vc_leafCert, vc_subCert2,
                        vc_subCert1));
            }
        }
    }
}

```



```

        } else {log("Invalid security type was chosen.");}
    }
}
return getVerdict;
}

function f_EVCC_CMN_PR_SupportedAppProtocol_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_SDP_001(v_HAL_61851_Listener);
    if ( verdict == pass ) {
        f_EVCC_CMN_PR_InitConfiguration_002(system, inconc);
        //clear SDP Port
        pt_V2G_UDP_SDP_Port.clear;
        verdict := f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001(inconc);
        if(verdict == pass) {
            vc_TcpPortIsMapped := false;
        }
        vc_Default_SDP_Message := activate(a_SDP_Message());
    }
    return verdict;
}

function f_EVCC_CMN_PR_SessionSetup_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);
    if ( verdict == pass ) {
        verdict := f_EVCC_CMN_TB_VTB_SessionSetup_001(inconc);
    }
    return verdict;
}

function f_EVCC_CMN_PR_ServiceDiscovery_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);
    if ( verdict == pass ) {
        verdict := f_EVCC_CMN_TB_VTB_ServiceDiscovery_001(inconc);
    }
    return verdict;
}

function f_EVCC_CMN_PR_ServiceDetail_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);
    if ( verdict == pass ) {
        if (PICS_CMN_CMN_IdentificationMode == pnC) {
            verdict := f_EVCC_CMN_TB_VTB_ServiceDetail_001(inconc); }
        if (PICS_CMN_CMN_IdentificationMode == eIM) {
            verdict := f_EVCC_CMN_TB_VTB_ServiceDetail_002(inconc); }
    }
    return verdict;
}

function f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(out HAL_61851_Listener
    v_HAL_61851_Listener)
    runs on EVCC_Tester
    return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);
    if ( verdict == pass ) {
        if (PICS_CMN_CMN_IdentificationMode == pnC) {
            verdict := f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_001(inconc); }
        if (PICS_CMN_CMN_IdentificationMode == eIM) {
            verdict := f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_002(inconc); }
    }
    return verdict;
}

function f_EVCC_CMN_PR_CertificateInstallation_001(out HAL_61851_Listener
    v_HAL_61851_Listener)
    runs on EVCC_Tester
    return verdicttype {

    var verdicttype verdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);
    if ( verdict == pass ) {

```

```

        verdict := f_EVCC_CMN_TB_VTB_CertificateInstallation_001(inconc);
    }
    return verdict;
}

function f_EVCC_CMN_PR_CertificateUpdate_001(out HAL_61851_Listener
                                             v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);
    if ( verdict == pass ) {
        verdict := f_EVCC_CMN_TB_VTB_CertificateUpdate_001(inconc);
    }
    return verdict;
}

function f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(out HAL_61851_Listener
                                                                v_HAL_61851_Listener)
    runs on EVCC_Tester
    return verdicttype {

    var verdicttype verdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);
    if ( verdict == pass ) {
        verdict := f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_001(inconc);
    }
    return verdict;
}

function f_EVCC_CMN_PR_Authorization_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict;
    if (PICS_CMN_CMN_IdentificationMode == pnC) { verdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener); }
    if (PICS_CMN_CMN_IdentificationMode == eIM) {
        verdict := f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener); }
    if ( verdict == pass ) {
        if (PICS_CMN_CMN_IdentificationMode == pnC) {
            verdict := f_EVCC_CMN_TB_VTB_Authorization_001(inconc); }
        if (PICS_CMN_CMN_IdentificationMode == eIM) {
            verdict := f_EVCC_CMN_TB_VTB_Authorization_002(inconc); }
    }
    return verdict;
}

function f_EVCC_CMN_PR_ChargeParameterDiscovery_001(out HAL_61851_Listener
                                                    v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);
    if ( verdict == pass ) {
        verdict := f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_001(inconc);
    }
    return verdict;
}

function f_EVCC_AC_PR_PowerDeliveryStart_001(out HAL_61851_Listener
                                             v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict :=
    f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);
    if ( verdict == pass ) {
        verdict := f_EVCC_AC_TB_VTB_PowerDelivery_001(v_HAL_61851_Listener, inconc);
    }
    return verdict;
}

function f_EVCC_AC_PR_ChargingStatus_001(out HAL_61851_Listener
                                           v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);
    if ( verdict == pass and not vc_sleepModeWoCharging) {
        verdict := f_EVCC_AC_TB_VTB_ChargingStatus_001(inconc);
    }
    return verdict;
}

```

```

}

function f_EVCC_AC_PR_MeteringReceipt_001(out HAL_61851_Listener
                                         v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);
    if ( verdict == pass and not vc_sleepModeWoCharging) {
        verdict := f_EVCC_AC_TB_VTB_MeteringReceipt_001(inconc);
    }
    return verdict;
}

function f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(out HAL_61851_Listener
                                                         v_HAL_61851_Listener)
    runs on EVCC_Tester
    return verdicttype {

    var verdicttype verdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);
    if ( verdict == pass and not vc_sleepModeWoCharging) {
        if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
            verdict := f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_002(v_HAL_61851_Listener,
                                                                           inconc);
        }
        else {
            verdict := f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener,
                                                                           inconc);
        }
    }
    return verdict;
}

function f_EVCC_AC_PR_SessionStop_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict :=
    f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);
    if ( verdict == pass ) {
        verdict := f_EVCC_AC_TB_VTB_SessionStop_002(inconc);
    }
    return verdict;
}

function f_EVCC_AC_PR_SessionStop_002(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict :=
    f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);
    if ( verdict == pass ) {
        verdict := f_EVCC_AC_TB_VTB_SessionStop_001(inconc, pass, inconc);
    }
    return verdict;
}

function f_EVCC_AC_PR_SessionStop_003(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict :=
    f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);
    if ( verdict == pass ) {
        verdict := f_EVCC_AC_TB_VTB_SessionStop_001(pass, inconc, inconc);
    }
    return verdict;
}

// Renegotiation
function f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001(out HAL_61851_Listener
                                                                      v_HAL_61851_Listener)
    runs on EVCC_Tester
    return verdicttype {

    var verdicttype verdict;
    if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
        verdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);
    }
    else {verdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);}

    if (verdict == pass){
        if (PICS_CMN_CMN_Renegotiation and PIXIT_CMN_CMN_Renegotiation == unknown) {

```

```

        if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
            verdict := f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_006(inconc);
        }
        else {verdict := f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_005(inconc);}
    }
    else if (not PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == renegotiation) {
        if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
            verdict := f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_009(inconc);
        }
        else {verdict := f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_010(inconc);}
    }
}
return verdict;
}

// Renegotiation
function f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(out HAL_61851_Listener
    v_HAL_61851_Listener)
    runs on EVCC_Tester
    return verdicttype {

    var verdicttype verdict;
    verdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001(
        v_HAL_61851_Listener);

    if (verdict == pass){
        verdict := f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_003(inconc);
    }
    return verdict;
}

// PreConditions for DC-TestCases
function f_EVCC_DC_PR_CableCheck_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);
    if (verdict == pass){
        verdict := f_EVCC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, inconc);
    }
    return verdict;
}

function f_EVCC_DC_PR_PreCharge_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict;
    verdict := f_EVCC_DC_PR_CableCheck_001(v_HAL_61851_Listener);
    if (verdict == pass){
        verdict := f_EVCC_DC_TB_VTB_PreCharge_001(inconc);
    }
    return verdict;
}

function f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(out HAL_61851_Listener
    v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);
    if (verdict == pass){
        verdict := f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_001(inconc);
    }
    return verdict;
}

function f_EVCC_DC_PR_CurrentDemand_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict :=
        f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);
    if (verdict == pass and not vc_sleepModeWoCharging){
        verdict := f_EVCC_DC_TB_VTB_CurrentDemand_001(inconc);
    }
    return verdict;
}

function f_EVCC_DC_PR_MeteringReceipt_001(out HAL_61851_Listener v_HAL_61851_Listener)
    runs on EVCC_Tester return verdicttype {

```

```

var verdicttype verdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);
if ( verdict == pass and not vc_sleepModeWoCharging) {
    verdict := f_EVCC_DC_TB_VTB_MeteringReceipt_001(inconc);
}
return verdict;
}

function f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(out HAL_61851_Listener
v_HAL_61851_Listener)
runs on EVCC_Tester
return verdicttype {

var verdicttype verdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

if (verdict == pass and not vc_sleepModeWoCharging){
    if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
        verdict := f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_002(inconc);
    }
    else {verdict := f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_001(inconc);}
}
return verdict;
}

function f_EVCC_DC_PR_WeldingDetection_001(out HAL_61851_Listener v_HAL_61851_Listener)
runs on EVCC_Tester return verdicttype {

var verdicttype verdict :=
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);
if (verdict == pass){
    verdict := f_EVCC_DC_TB_VTB_WeldingDetection_001(v_HAL_61851_Listener, inconc);
}
return verdict;
}

function f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001(out HAL_61851_Listener
v_HAL_61851_Listener)
runs on EVCC_Tester return verdicttype {

var verdicttype verdict :=
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);
if (verdict == pass){
    verdict := f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_003(v_HAL_61851_Listener,
inconc);
}
return verdict;
}

function f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(out HAL_61851_Listener
v_HAL_61851_Listener)
runs on EVCC_Tester return verdicttype {

var verdicttype verdict :=
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);
if (verdict == pass){
    verdict := f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
inconc, pass, inconc);
}
return verdict;
}

function f_EVCC_DC_PR_WeldingDetectionOrSessionStop_003(out HAL_61851_Listener
v_HAL_61851_Listener)
runs on EVCC_Tester return verdicttype {

var verdicttype verdict :=
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);
if (verdict == pass){
    verdict := f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
pass, inconc, inconc);
}
return verdict;
}

// Renegotiation
function f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001(out HAL_61851_Listener
v_HAL_61851_Listener)
runs on EVCC_Tester
return verdicttype {

```

```

var verdicttype verdict;
if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
    verdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);
}
else {verdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);}

if (verdict == pass){
    if (PICS_CMN_CMN_Renegotiation and PIXIT_CMN_CMN_Renegotiation == unknown) {
        if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
            verdict := f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_006(inconc);
        }
        else {verdict := f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_005(inconc);}
    }
    else if (not PICS_CMN_CMN_Renegotiation and
        PIXIT_CMN_CMN_Renegotiation == renegotiation) {
        if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
            verdict := f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_009(inconc);
        }
        else {verdict := f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_010(inconc);}
    }
}
return verdict;
}

// Renegotiation
function f_EVCC_DC_PR_PreChargeRenegotiation_001(out HAL_61851_Listener
    v_HAL_61851_Listener)
    runs on EVCC_Tester
    return verdicttype {

var verdicttype verdict;
verdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001(
    v_HAL_61851_Listener);

if (verdict == pass){
    verdict := f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_003(inconc);
}

if (verdict == pass){
    verdict := f_EVCC_DC_TB_VTB_CableCheck_007(v_HAL_61851_Listener, inconc);
}

if (verdict == pass){
    verdict := f_EVCC_DC_TB_VTB_PreCharge_001(inconc);
}
return verdict;
}

// Sleep mode
function f_EVCC_CMN_PR_PausedSession_001(out HAL_61851_Listener
    v_HAL_61851_Listener)
    runs on EVCC_Tester
    return verdicttype {

var verdicttype verdict := pass;
var float v_time;
timer statetimer;

v_HAL_61851_Listener.stop;
v_HAL_61851_Listener.start(f_EVCC_HAL61851Listener(true));

if(PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    f_EVCC_setPwmMode(e_OscOn);
    f_EVCC_setDutyCycle(5);
} else {
    // BCB toggle sequence detection
    f_EVCC_changeValidStateCondition(B,C);
    if((PICS_EVCC_CMN_PmaxScheduleWithZeroPow != none_) and
        (par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp)){

        v_time := (PICS_CMN_CMN_WakeUp - par_SECC_Pmax0W + 5.0);
    } else {
        v_time := (PICS_CMN_CMN_WakeUp - PIXIT_CMN_CMN_WakeUp + 5.0);
    }
    statetimer.start(v_time);
    alt {
        [] a_EVCC_BCBToggleDetection(pt_HAL_61851_Internal_Port, B){
            statetimer.stop;
        }
    }
}
}

```

```

[] a_EVCC_BCBToggleDetection(pt_HAL_61851_Internal_Port, C){
    statetimer.stop;
    f_EVCC_changeValidStateCondition(C,B);
    statetimer.start(par_T_vald_state_duration_max);
    repeat;
}
[] pt_HAL_61851_Internal_Port.receive {
    setverdict(inconc, "Received state has an invalid value.");
}
[] statetimer.timeout {
    setverdict(inconc, "The EVSE could not detect the corresponding " &
        "toggle value within " &
        "the maximal valid state duration.");
}
}
if(getverdict != pass) {
    log("The SUT did not initiate a wake-up " &
        "within 'PICS_CMN_CMN_WakeUp'.");
}
}

f_EVCC_getPLCLinkEstablishmentAfterSleepMode(inconc);
unmap(mtc:pt_SLAC_Port, system:pt_SLAC_Port);

tc_V2G_SECC_CommunicationSetup_Timer.start;
return getverdict;
}

function f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(out HAL_61851_Listener
    v_HAL_61851_Listener)
    runs on EVCC_Tester
    return verdicttype {

var verdicttype verdict := pass;
var float v_time;
timer statetimer;

v_HAL_61851_Listener.stop;
v_HAL_61851_Listener.start(f_EVCC_HAL61851_Listener(true));

if(PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
    f_EVCC_setPwmMode(e_OscOn);
    f_EVCC_setDutyCycle(5);
} else {
    // BCB toggle sequence detection
    f_EVCC_changeValidStateCondition(B,C);
    if((PICS_EVCC_CMN_PmaxScheduleWithZeroPow != none_) and
        (par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp)){
        v_time := (PICS_CMN_CMN_WakeUp - par_SECC_Pmax0W + 5.0);
    } else {
        v_time := (PICS_CMN_CMN_WakeUp - PIXIT_CMN_CMN_WakeUp + 5.0);
    }
    statetimer.start(v_time);
    alt {
        [] a_EVCC_BCBToggleDetection(pt_HAL_61851_Internal_Port, B){
            statetimer.stop;
        }
        [] a_EVCC_BCBToggleDetection(pt_HAL_61851_Internal_Port, C){
            statetimer.stop;
            f_EVCC_changeValidStateCondition(C,B);
            statetimer.start(par_T_vald_state_duration_max);
            repeat;
        }
        [] pt_HAL_61851_Internal_Port.receive {
            setverdict(inconc, "Received state has an invalid value.");
        }
        [] statetimer.timeout {
            setverdict(inconc, "The EVSE could not detect the corresponding " &
                "toggle value within " &
                "the maximal valid state duration.");
        }
    }
}
if(getverdict != pass) {
    log("The SUT did not initiate a wake-up " &
        "within 'PICS_CMN_CMN_WakeUp'.");
}
}

```



```

    }
}

f_EVCC_getPLCLinkEstablishmentAfterSleepMode(inconc);
unmap(mtc:pt_SLAC_Port, system:pt_SLAC_Port);

if(getverdict == pass) {
    tc_V2G_SECC_CommunicationSetup_Timer.start;
    // EVCC_Binding Behavior
    if ( verdict == pass ) {
        var template Security_TYPE v_security := ?;
        verdict := f_EVCC_CMN_TB_VTB_SDP_001(v_security, inconc);
        if(PICS_CMN_CMN_IdentificationMode == pnC) {
            if(vc_Security != cc_hexTls){
                setverdict(inconc, "SUT did not request the correct " &
                    "transmission security.");
            }
        }
        if(PICS_CMN_CMN_IdentificationMode == eIM and
            PIXIT_EVCC_CMN_TLS == none_ ) {
            if(vc_Security != cc_hexTcp){
                setverdict(inconc, "SUT did not request the correct " &
                    "transmission security.");
            }
        }
        if(getverdict == pass) {
            if(vc_Security == cc_hexTcp) {
                vc_v2g_tcp_tls_parameter := md_CMN_CMN_TcpParameter_001(
                    vc_sut_ipAddress.ipAddress,
                    vc_V2G_Port_PortNumber,
                    vc_Security);

            } else if(vc_Security == cc_hexTls) {
                vc_v2g_tcp_tls_parameter := md_CMN_CMN_TlsParameter_001(
                    vc_sut_ipAddress.ipAddress,
                    vc_V2G_Port_PortNumber,
                    vc_Security, vc_cipherSuite,
                    vc_enabledProtocol,
                    md_CertificateChainType_002(
                        vc_leafCert, vc_subCert2,
                        vc_subCert1));
            } else {log("Invalid security type was chosen.");}
        }
    }
}

if ( verdict == pass ) {
    f_EVCC_CMN_PR_InitConfiguration_002(system, inconc);
    //clear SDP Port
    pt_V2G_UDP_SDP_Port.clear;
    verdict := f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001(inconc);
    if(verdict == pass) {
        vc_TcpPortIsMapped := false;
    }
    vc_Default_SDP_Message := activate(a_SDP_Message());
}
return verdict;
}

function f_EVCC_CMN_PR_SessionSetupPausedSession_001(out HAL_61851_Listener
    v_HAL_61851_Listener)
    runs on EVCC_Tester
    return verdicttype {

var verdicttype verdict;

verdict := f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(
    v_HAL_61851_Listener);

// SECC_SessionSetup Behavior
if ( verdict == pass ) {
    verdict := f_EVCC_CMN_TB_VTB_SessionSetup_006(inconc);
}
if ( verdict == pass ) {
    verdict := f_EVCC_CMN_TB_VTB_ServiceDiscovery_006(inconc);
}
return verdict;
}

function f_EVCC_CMN_PR_AuthorizationPausedSession_001(out HAL_61851_Listener

```

```

v_HAL_61851_Listener)
runs on EVCC_Tester
return verdicttype {

var verdicttype verdict;

verdict := f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);

// SECC_SessionSetup Behavior

if ( verdict == pass ) {
  if (PICS_CMN_CMN_IdentificationMode == pnC) {
    verdict := f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(inconc); }
  if (PICS_CMN_CMN_IdentificationMode == eIM) {
    verdict := f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(inconc); }
}
if ( verdict == pass ) {
  if (PICS_CMN_CMN_IdentificationMode == pnC) {
    verdict := f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_001(inconc); }
}
if ( verdict == pass ) {
  if (PICS_CMN_CMN_IdentificationMode == pnC) {
    verdict := f_EVCC_CMN_TB_VTB_Authorization_001(inconc); }
  if (PICS_CMN_CMN_IdentificationMode == eIM) {
    verdict := f_EVCC_CMN_TB_VTB_Authorization_002(inconc); }
}
return verdict;
}
}

```

F.3 Post-condition functions

F.3.1 SECC functions

```

module PostConditions_SECC_15118_2 {

import from Services_HAL_61851 all;
import from ComponentsAndPorts all;
import from DataStructure_HAL_61851 all;
import from LibFunctions_15118_2 { group generalFunctions; }
import from Timer_15118_2 all;
import from Timer_15118_3 {modulepar par_CMN_waitForKeyReset;};
import from DataStructure_V2G_TCP_TLS_Port_Control all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from TestBehavior_SECC_CommonBehavior all;
import from Pics_15118 all;
import from Pixit_15118 all;
import from Timer_15118 all;

function f_SECC_CMN_PO_InitialState_001(out HAL_61851_Listener v_HAL_61851_Listener)
runs on SECC_Tester

{
  v_HAL_61851_Listener.stop;
  v_HAL_61851_Listener.start(f_SECC_HAL61851Listener(false));

  var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_internalResponseMessage;
  var boolean isClosed := false;
  var V2G_TCP_TLS_Port_Control_Ack_TYPE ackType;

  pt_V2G_TCP_TLS_ALM_SECC_Port.clear;
  deactivate(vc_Default_IEC_61851_ListenerBehavior);
  deactivate(vc_Default_SDP_Message);

  if(pt_V2G_TCP_TLS_ALM_SECC_Port.checkstate("Mapped") and not(vc_isTcpPortClosed)){
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
      e_disconnectPort)) to "local";
    alt{
      [pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_CMN_tcpTlsStatusRes_001(
        e_disconnectPort, ?, ?))
        -> value v_internalResponseMessage {

        isClosed := v_internalResponseMessage.result.portStatus.closed;
        ackType := v_internalResponseMessage.ack;
      }
    }
  }
}
}

```

```

    pt_HAL_61851_Internal_Port.clear;
    pt_HAL_61851_Port.clear;
    f_SECC_changeValidStateCondition(A);
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    f_SECC_setState(A,v_HAL_61851_Listener);
    pt_HAL_61851_Port.clear;
    sleep((par_CMN_waitForNextHAL));
    f_SECC_setProximity(0);
    v_HAL_61851_Listener.stop;

    all timer.stop;

    log((par_SECC_waitForNextTC)," Sec timer started");
    sleep((par_SECC_waitForNextTC));
    log((par_SECC_waitForNextTC)," Sec timer stopped");
}

```

F.3.2 EVCC functions

```

module PostConditions_EVCC_15118_2 {

    import from Services_HAL_61851 all;
    import from ComponentsAndPorts all;
    import from DataStructure_HAL_61851 all;
    import from LibFunctions_15118_2 all;
    import from Pics_15118 all;
    import from Timer_15118_2 all;
    import from DataStructure_V2G_TCP_TLS_Port_Control all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Pixit_15118 all;
    import from Timer_15118 all;

    function f_EVCC_CMN_PO_InitialState_001(out HAL_61851_Listener v_HAL_61851_Listener)
        runs on EVCC_Tester {

        v_HAL_61851_Listener.stop;
        v_HAL_61851_Listener.start(f_EVCC_HAL61851Listener(false));

        var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_internalResponseMessage;
        var boolean isClosed := false;
        var V2G_TCP_TLS_Port_Control_Ack_TYPE ackType;

        pt_V2G_TCP_TLS_ALM_EVCC_Port.clear;
        deactivate(vc_Default_IEC_61851_ListenerBehavior);
        deactivate(vc_Default_SDP_Message);

        if(vc_TcpPortIsMapped and not(vc_isTcpPortClosed)){
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                e_disconnectPort)) to "local";
            alt{
                [pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_CMN_tcpTlsStatusRes_001(
                    e_disconnectPort,?,?))
                    -> value v_internalResponseMessage {

                    isClosed := v_internalResponseMessage.result.portStatus.closed;
                    ackType := v_internalResponseMessage.ack;

                }
            }
        }

        pt_HAL_61851_Port.clear;
        f_EVCC_setPwmMode(e_OscOff);
        pt_HAL_61851_Port.clear;
        sleep((par_CMN_waitForNextHAL));
        f_EVCC_setProximity(0);
        v_HAL_61851_Listener.stop;

        all timer.stop;

        log((par_EVCC_waitForNextTC)," Sec timer started");

```

```

    sleep((par_EVCC_waitForNextTC));
    log((par_EVCC_waitForNextTC), " Sec timer stopped");
}
}

```

F.4 Common behavior functions

F.4.1 SECC functions

```

module TestBehavior_SECC_CommonBehavior {

import from DataStructure_HAL_61851 all;
import from Templates_CMN_HAL61851 all;
import from Templates_CMN_V2GMessage all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from ComponentsAndPorts all;
import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
import from urn_iso_15118_2_2013_MsgDef all;
import from DataStructure_SDP all;
import from Services_PLCLinkStatus all;

altstep a_SECC_Unexpected_Message_Content(verdicttype v_verdict, charstring v_message)
    runs on SECC_Tester {

    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive
    {
        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(v_verdict, v_message);
    }
}

altstep a_SECC_Unexpected_Message(verdicttype v_verdict, charstring v_message)
    runs on SECC_Tester {

    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive
    {
        setverdict(v_verdict, v_message);
    }
}

altstep a_SECC_Unexpected_V2G_Message(verdicttype v_verdict, charstring v_message,
    template BodyElement_group body)
    runs on SECC_Tester {

    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(?
        md_CMN_BodyType_001(complement(body)))) {
        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(v_verdict, v_message);
    }
}

altstep a_SECC_Timeout(verdicttype v_verdict) runs on SECC_Tester {

    [] tc_V2G_EVCC_Msg_Timer.timeout {
        setverdict(v_verdict, "Nothing received. V2G_EVCC_Msg_Timer " &
            "was triggered.");
    }
}

altstep a_No_SDP_Message(charstring v_message, verdicttype v_verdict)
    runs on SECC_Tester {

    [] pt_V2G_UDP_SDP_Port.receive {
        tc_EVCC_SDP_Timer.stop;
        setverdict(v_verdict, v_message);
    }
}

altstep a_SECC_SDP_Timeout(verdicttype v_verdict) runs on SECC_Tester {

    [] tc_EVCC_SDP_Timer.timeout {
        setverdict(v_verdict, "Nothing received. tc_EVCC_SDP_Timer " &
            "was triggered.");
    }
}

```

```

}

altstep a_SDP_Message() runs on SECC_Tester {

    var Sender_IPaddress_TYPE v_sut_ipAddress;

    [] pt_V2G_UDP_SDP_Port.receive(SDP_Message:?)
    -> sender v_sut_ipAddress.ipAddress {

        if(v_sut_ipAddress.ipAddress == vc_sut_ipAddress.ipAddress) {
            setverdict(fail, "SDPMessage was received, but not expected.");
        }
        else {
            repeat;
        }
    }
}

altstep a_SECC_CommunicationSetup_Timeout(verdicttype v_verdict)
    runs on SECC_Tester {

    [] tc_V2G_EVCC_CommunicationSetup_Timer.timeout {

        setverdict(v_verdict, "Nothing received. V2G_EVCC_CommunicationSetup_Timer " &
            "was triggered.");
    }
}

altstep a_SECC_TCPConnection_Status_Listener(verdicttype v_verdict, charstring v_message)
    runs on SECC_Tester {

    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_CMN_tcpTlsStatusRes_001(
        e_disconnectPort, e_success, true)) {

        setverdict(v_verdict, v_message);
        vc_isTcpPortClosed := true;
    }
}

altstep a_SECC_TCP_Connection_Termination_Timeout() runs on SECC_Tester {

    [] tc_CMN_TCP_Connection_Termination_Timer.timeout {
        setverdict(fail, "TCP connection was >>NOT<< terminated by the SUT, " &
            "tc_CMN_TCP_Connection_Termination_Timer was triggered.");
    }
}

function isError() return boolean {

    if(getverdict == error) {
        return true;
    }
    return false;
}

function f_SECC_startSleepingPhase(in float v_time) runs on SECC_Tester {

    timer t_sleep := v_time;
    t_sleep.start;
    alt {
        [] t_sleep.timeout {
            log("Sleeping phase was terminated by the Test System.")
        }
        [] a_SECC_processPLCLinkNotifications_001();
        [] pt_SLAC_Port.receive
        {
            setverdict(fail, "Invalid message was received within " &
                "sleeping phase on port 'pt_SLAC_Port'.");
        }
        [] pt_V2G_UDP_SDP_Port.receive
        {
            setverdict(fail, "Invalid message was received within " &
                "sleeping phase on port 'pt_V2G_UDP_SDP_Port'.");
        }
    }
}
}

```

F.4.2 EVCC functions

```

module TestBehavior_EVCC_CommonBehavior {

    import from DataStructure_HAL_61851 all;
    import from Templates_CMN_HAL61851 all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Templates_CMN_V2GMessage all;
    import from ComponentsAndPorts all;
    import from DataStructure_V2G_TCP_TLS_Port_Control all;
    import from urn_iso_15118_2_2013_MsgBody all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from DataStructure_SDP all;
    import from Services_PLCLinkStatus all;

    altstep a_EVCC_Unexpected_Message_Content(verdicttype v_verdict, charstring v_message)
        runs on EVCC_Tester {

        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive
        {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(v_verdict, v_message);
        }
    }

    altstep a_EVCC_Unexpected_Message(verdicttype v_verdict, charstring v_message)
        runs on EVCC_Tester {

        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive
        {
            setverdict(v_verdict, v_message);
        }
    }

    altstep a_EVCC_Unexpected_V2G_Message(verdicttype v_verdict, charstring v_message,
        template BodyElement_group body)
        runs on EVCC_Tester {

        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(?
            md_CMN_BodyType_001(complement(body)))) {

            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(v_verdict, v_message);
        }
    }

    altstep a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(verdicttype v_verdict, charstring v_message,
        template BodyElement_group body1,
        template BodyElement_group body2)
        runs on EVCC_Tester {

        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(?
            md_CMN_BodyType_001(complement(body1, body2)))) {

            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(v_verdict, v_message);
        }
    }

    altstep a_EVCC_Unexpected_V2G_Message_PIXIT_Var3(verdicttype v_verdict, charstring v_message,
        template BodyElement_group body1,
        template BodyElement_group body2,
        template BodyElement_group body3)
        runs on EVCC_Tester {

        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(?
            md_CMN_BodyType_001(complement(body1, body2, body3)))) {

            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(v_verdict, v_message);
        }
    }

    altstep a_EVCC_Timeout(verdicttype v_verdict) runs on EVCC_Tester {

        [] tc_V2G_SECC_Sequence_Timer.timeout
        {
    
```

```

        setverdict(v_verdict, "Nothing received. V2G_EVCC_Msg_Timer was triggered.");
    }
}

altstep a_EVCC_TCPConnection_Status_Listener(verdicttype v_verdict,
        charstring v_message)
    runs on EVCC_Tester {

    []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_CMN_tcpTlsStatusRes_001(e_disconnectPort,
        e_success, true)) {
        setverdict(v_verdict,v_message);
        vc_isTcpPortClosed := true;
    }
}

altstep a_EVCC_No_SDP_Message(charstring v_message, verdicttype v_verdict)
    runs on EVCC_Tester {

    [] pt_V2G_UDP_SDP_Port.receive
    {
        tc_V2G_SECC_CommunicationSetup_Timer.stop;
        setverdict(v_verdict, v_message);
    }
}

altstep a_EVCC_SDP_Timeout(verdicttype v_verdict) runs on EVCC_Tester {

    [] tc_V2G_SECC_CommunicationSetup_Timer.timeout {
        setverdict(v_verdict, "Nothing received. tc_EVCC_CommunicationSetup_Timer " &
            "was triggered.");
    }
}

altstep a_SDP_Message() runs on EVCC_Tester {

    var Sender_IPAddress_TYPE v_sut_ipAddress;

    [] pt_V2G_UDP_SDP_Port.receive(SDP_Message:?)
    -> sender v_sut_ipAddress.ipAddress {

        if(v_sut_ipAddress.ipAddress == vc_sut_ipAddress.ipAddress) {
            setverdict(fail, "SDPMessage was received, but not expected.");
        }
        else {
            repeat;
        }
    }
}

altstep a_EVCC_TCP_Connection_Termination_Timeout() runs on EVCC_Tester {

    [] tc_CMN_TCP_Connection_Termination_Timer.timeout {
        setverdict(fail, "TCP connection was >>NOT<< terminated by the SUT, " &
            "tc_CMN_TCP_Connection_Termination_Timer was triggered.");
    }
}

function isError() return boolean {
    if(getverdict == error) {
        return true;
    }
    return false;
}

function f_EVCC_startSleepingPhase(in float v_time) runs on EVCC_Tester {

    timer t_sleep := v_time;

    t_sleep.start;
    alt {
        [] t_sleep.timeout {
            log("Sleeping phase was terminated by the Test System.")
        }
        [] a_EVCC_processPLCLinkNotifications_001();
        [] pt_SLAC_Port.receive
        {
            setverdict(fail, "Invalid message was received within " &

```



```

        "sleeping phase on port 'pt_SLAC_Port'.");
    }
    [] pt_V2G_UDP_SDP_Port.receive
    {
        setverdict(fail, "Invalid message was received within " &
            "sleeping phase on port 'pt_V2G_UDP_SDP_Port'.");
    }
}
}
}

```

F.5 Library functions

```

module LibFunctions_15118_2 {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from XSDAUX all;
    import from Templates_CMN_Security all;
    import from urn_iso_15118_2_2013_MsgHeader language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" {type V2G_Message;};
    import from urn_iso_15118_2_2013_MsgBody all;
    import from Templates_CMN_V2GMessage all;
    import from DataStructure_Security all;
    import from http_www_w3_org_2000_09_xmldsig language "XSD" {
        type Signature, SignatureType, SignedInfo, ReferenceType;
    }
    import from Services_Security all;
    import from urn_iso_15118_2_2013_X509Def all;
    import from Pics_15118_2 all;
    import from TTlibrary all;

    group PhysicalValues {

        function f_CMN_PhysicalValue_GetValue(in PhysicalValueType v) return float {
            var float ret;
            if (v.multiplier < 0) { ret := (int2float(v.value_) * (1.0/(f_CMN_Pow(v.multiplier*-1))))};
            else if (v.multiplier > 0) { ret := (int2float(v.value_) * (f_CMN_Pow(v.multiplier)))};
            else { ret := int2float(v.value_); }

            return ret;
        }

        function f_CMN_Pow(in integer v_multiplier) return float {
            var float ret := 10.0;
            for (var integer i:=0; i<v_multiplier-1; i:=i + 1) {
                ret := ret * 10.0;
            }
            return ret;
        }

        function f_CMN_AsbFlt(in float v_first, in float v_second) return float {
            if(v_first == v_second){
                return 0.0;
            }
            if(v_first > v_second){
                return v_first - v_second;
            }else{
                return v_second - v_first;
            }
        }

        function f_CMN_AsbInt(in integer v_first, in integer v_second) return integer {
            if(v_first == v_second){
                return 0;
            }
            if(v_first > v_second){
                return v_first - v_second;
            }else{
                return v_second - v_first;
            }
        }

        function f_CMN_Float2Int(in float v_float, in integer v_position) return integer {

            for (var integer i:=0; i<v_position; i:=i + 1) {
                v_float := v_float * 10.0;
            }
        }
    }
}

```

```

    }
    return float2int(v_float);
}

function f_CMN_Sqrt(in float v_Float) return float {

    var float v_x := 1.0;
    var float v_xAlt := 0.0;
    var float v_epsilon := 0.00001;

    while (f_CMN_AsbFlt(v_x, v_xAlt) > v_epsilon) {
        v_xAlt := v_x;
        v_x := (v_x * v_x + v_Float) / (2.0 * v_x);
    }

    return v_x;
}

function f_check_PhysicalValue_valueRange(in PhysicalValueType v_physicalValue,
    in UnitSymbolType v_unit,
    in float v_maxValue,
    in verdicttype v_vct) {

    if(f_CMN_PhysicalValue_GetValue(v_physicalValue) > v_maxValue) {
        setverdict(v_vct, "The corresponding physical value has an invalid maximum value.");
    }

    if(v_physicalValue.unit != v_unit) {
        setverdict(v_vct, "The corresponding physical value has an invalid unit.");
    }
}

}

group generalFunctions {

    type record ListOfFloat {
        record length (0 .. 10) of float v_float
    }

    type record ListOfBoolean {
        record length (0 .. 10) of boolean v_boolean
    }

    external function fx_logToFile(in template ListOfFloat v_listOfFloat,
        in template ListOfBoolean v_listOfBoolean);

    external function fx_captureTraffic(in integer v_interfaceIdx);

    external function fx_stopCapturing();

    external function fx_validateEVSEID(in EvseIDType v_evseID,
        in charstring v_protocol) return boolean;

    external function fx_validateEMAID(in charstring v_eMAID) return boolean;

    external function fx_checkCertificateValidity(in charstring v_notBefore,
        in charstring v_notAfter)
        return boolean;

    external function fx_validateCertID(in charstring v_certID) return boolean;

    function sleep(float time) {
        timer t := time;
        t.start;
        t.timeout;
    }
}

group security {

    function f_generateDigestFromBodyType(in template (value) BodyType p_BodyType)
        return XSDAUX.base64Binary {
            return fx_convertOctetstring2Base64Binary(fx_messageDigest(
                "SHA-256", fx_canonicalExi(
                    encvalue(p_BodyType))));
        }

    function f_calculateIdDigestMapFromBodyType(in template (value) BodyType p_BodyType)
        return idDigestValueMap {
            var idDigestValueMap resultIdDigestValueMap := {};

```

```

if (ischosen(p_BodyType.bodyElement.authorizationReq)) {
    var charstring hashedExiHexstringAsBase64BinaryString :=
        fx_convertOctetstring2Base64Binary(fx_messageDigest("SHA-256",
            fx_canonicalExi(encvalue(p_BodyType))));

    resultIdDigestValueMap := {
        [0] := {p_BodyType.bodyElement.authorizationReq.id,
            hashedExiHexstringAsBase64BinaryString}
    };
} else if (ischosen(p_BodyType.bodyElement.meteringReceiptReq)) {
    var charstring hashedExiHexstringAsBase64BinaryString :=
        fx_convertOctetstring2Base64Binary(fx_messageDigest("SHA-256",
            fx_canonicalExi(encvalue(p_BodyType))));

    resultIdDigestValueMap := {
        [0] := {p_BodyType.bodyElement.meteringReceiptReq.id,
            hashedExiHexstringAsBase64BinaryString}
    };
} else if (ischosen(p_BodyType.bodyElement.chargeParameterDiscoveryRes)) {
    var ChargeParameterDiscoveryRes cpdRes := p_BodyType.bodyElement.
        chargeParameterDiscoveryRes;

    if (ispresent(cpdRes.sASchedules)) {
        var SASchedules_group schedules := cpdRes.sASchedules;
        if (ischosen(schedules.sAScheduleList)) {
            var integer v_counter;
            var integer v_counterIdDigestValueMap := 0;
            var SAScheduleList list := schedules.sAScheduleList;
            for (v_counter := 0; v_counter < lengthof(list.sAScheduleTuple_list);
                v_counter := v_counter + 1) {
                var SAScheduleTupleType element := list.sAScheduleTuple_list[v_counter];

                if (ispresent(element.salesTariff)) {
                    var SalesTariffType currentSalesTariffType := element.salesTariff;
                    var octetstring salesTariffAsOctetstring :=
                        fx_marshallTtcn3V2GSalesTariffTypeWithJaxb(currentSalesTariffType);

                    var bitstring salesTariffAsBitString :=
                        oct2bit(salesTariffAsOctetstring);
                    var octetstring exiEncodedSalesTariff :=
                        fx_canonicalExi(salesTariffAsBitString);

                    resultIdDigestValueMap[v_counterIdDigestValueMap] := {
                        currentSalesTariffType.id,
                        fx_convertOctetstring2Base64Binary(
                            fx_messageDigest("SHA-256", exiEncodedSalesTariff)) };

                    v_counterIdDigestValueMap := v_counterIdDigestValueMap + 1;
                }
            }
        }
    }
} else if (ischosen(p_BodyType.bodyElement.certificateInstallationReq)) {
    var charstring hashedExiHexstringAsBase64BinaryString :=
        fx_convertOctetstring2Base64Binary(fx_messageDigest("SHA-256",
            fx_canonicalExi(encvalue(p_BodyType))));

    resultIdDigestValueMap := {
        [0] := {p_BodyType.bodyElement.certificateInstallationReq.id,
            hashedExiHexstringAsBase64BinaryString}
    };
} else if (ischosen(p_BodyType.bodyElement.certificateInstallationRes)) {
    var CertificateInstallationRes v_certificateInstallationRes :=
        p_BodyType.bodyElement.certificateInstallationRes;
    var CertificateChainType v_contractSignatureCertChain :=
        v_certificateInstallationRes.contractSignatureCertChain;
    var ContractSignatureEncryptedPrivateKeyType
        v_contractSignatureEncryptedPrivateKey :=
        v_certificateInstallationRes.contractSignatureEncryptedPrivateKey;
    var DiffieHellmanPublicKeyType v_dHpublickey :=
        v_certificateInstallationRes.dHpublickey;
    var EMAIDType v_eMAID := v_certificateInstallationRes.eMAID;

    var integer v_counterIdDigestValueMap := 0;
    var octetstring certificateChainTypeAsOctetstring :=
        fx_marshallTtcn3CertificateChainTypeWithJaxb(
            v_contractSignatureCertChain);
    var bitstring encodedContractSignatureCertChain :=
        oct2bit(certificateChainTypeAsOctetstring);

    resultIdDigestValueMap[v_counterIdDigestValueMap] := {
        v_contractSignatureCertChain.id,

```

```

        fx_convertOctetstring2Base64Binary(
            fx_messageDigest("SHA-256",
                fx_canonicalExi(
                    encodedContractSignatureCertChain))) );

v_counterIdDigestValueMap := v_counterIdDigestValueMap+1;

var octetstring contractSignatureEncryptedPrivateKeyTypeAsOctetstring :=
fx_marshallTtcn3ContractSignatureEncryptedPrivateKeyTypeWithJaxb(
v_contractSignatureEncryptedPrivateKey);
var bitstring encodedContractSignatureEncryptedPrivateKeyTypeAsBitstring :=
oct2bit(contractSignatureEncryptedPrivateKeyTypeAsOctetstring);

resultIdDigestValueMap[v_counterIdDigestValueMap] := {
    v_contractSignatureEncryptedPrivateKey.id,
    fx_convertOctetstring2Base64Binary(fx_messageDigest(
        "SHA-256", fx_canonicalExi(
            encodedContractSignatureEncryptedPrivateKeyTypeAsBitstring))) );

v_counterIdDigestValueMap := v_counterIdDigestValueMap+1;

var octetstring diffieHellmanPublicKeyTypeAsOctetstring :=
fx_marshallTtcn3DiffieHellmanPublicKeyTypeWithJaxb(v_dHpublickey);
var bitstring encodedDiffieHellmanPublicKeyTypeAsBitstring :=
oct2bit(diffieHellmanPublicKeyTypeAsOctetstring);

resultIdDigestValueMap[v_counterIdDigestValueMap] := {
    v_dHpublickey.id,
    fx_convertOctetstring2Base64Binary(
        fx_messageDigest("SHA-256",
            fx_canonicalExi(
                encodedDiffieHellmanPublicKeyTypeAsBitstring))) );

v_counterIdDigestValueMap := v_counterIdDigestValueMap+1;

var octetstring eMAIDTypeAsOctetstring :=
fx_marshallTtcn3EMAIDTypeWithJaxb(v_eMAID);
var bitstring eMAIDTypeAsBitstring := oct2bit(eMAIDTypeAsOctetstring);

resultIdDigestValueMap[v_counterIdDigestValueMap] := {
    v_eMAID.id,
    fx_convertOctetstring2Base64Binary(fx_messageDigest(
        "SHA-256", fx_canonicalExi(eMAIDTypeAsBitstring))) );

} else if (ischosen(p_BodyType.bodyElement.certificateUpdateReq)) {
    var charstring hashedExiHexstringAsBase64BinaryString :=
fx_convertOctetstring2Base64Binary(fx_messageDigest("SHA-256",
        fx_canonicalExi(
            encvalue(p_BodyType))));

    resultIdDigestValueMap := {
        [0] := {p_BodyType.bodyElement.certificateUpdateReq.id,
            hashedExiHexstringAsBase64BinaryString}
    };

} else if (ischosen(p_BodyType.bodyElement.certificateUpdateRes)) {
    var CertificateUpdateRes v_certificateUpdateRes :=
p_BodyType.bodyElement.certificateUpdateRes;
    var CertificateChainType v_contractSignatureCertChain :=
v_certificateUpdateRes.contractSignatureCertChain;
    var ContractSignatureEncryptedPrivateKeyType
v_contractSignatureEncryptedPrivateKey :=
v_certificateUpdateRes.contractSignatureEncryptedPrivateKey;
    var DiffieHellmanPublicKeyType v_dHpublickey := v_certificateUpdateRes.
        dHpublickey;

    var EMAIDType v_eMAID := v_certificateUpdateRes.eMAID;
    var integer v_counterIdDigestValueMap :=0;

    var octetstring certificateChainTypeAsOctetstring :=
fx_marshallTtcn3CertificateChainTypeWithJaxb(v_contractSignatureCertChain);
    var bitstring encodedContractSignatureCertChain :=
oct2bit(certificateChainTypeAsOctetstring);

    resultIdDigestValueMap[v_counterIdDigestValueMap] := {
        v_contractSignatureCertChain.id,
        fx_convertOctetstring2Base64Binary(
            fx_messageDigest("SHA-256",
                fx_canonicalExi(encodedContractSignatureCertChain))) );

    v_counterIdDigestValueMap := v_counterIdDigestValueMap+1;

```

```

var octetstring contractSignatureEncryptedPrivateKeyTypeAsOctetstring :=
fx_marshallTtcn3ContractSignatureEncryptedPrivateKeyTypeWithJaxb(
v_contractSignatureEncryptedPrivateKey);
var bitstring encodedContractSignatureEncryptedPrivateKeyTypeAsBitstring :=
oct2bit(contractSignatureEncryptedPrivateKeyTypeAsOctetstring);

resultIdDigestValueMap[v_counterIdDigestValueMap] := {
    v_contractSignatureEncryptedPrivateKey.id,
    fx_convertOctetstring2Base64Binary(
    fx_messageDigest("SHA-256",
    fx_canonicalExi(
    encodedContractSignatureEncryptedPrivateKeyTypeAsBitstring))) };

v_counterIdDigestValueMap := v_counterIdDigestValueMap+1;

var octetstring diffieHellmanPublicKeyTypeAsOctetstring :=
fx_marshallTtcn3DiffieHellmanPublicKeyTypeWithJaxb(v_dHpublickey);
var bitstring encodedDiffieHellmanPublicKeyTypeAsBitstring :=
oct2bit(diffieHellmanPublicKeyTypeAsOctetstring);

resultIdDigestValueMap[v_counterIdDigestValueMap] := {
    v_dHpublickey.id,
    fx_convertOctetstring2Base64Binary(
    fx_messageDigest("SHA-256",
    fx_canonicalExi(
    encodedDiffieHellmanPublicKeyTypeAsBitstring))) };

v_counterIdDigestValueMap := v_counterIdDigestValueMap+1;

var octetstring eMAIDTypeAsOctetstring :=
fx_marshallTtcn3EMAIDTypeWithJaxb(v_eMAID);
var bitstring eMAIDTypeAsBitstring := oct2bit(eMAIDTypeAsOctetstring);

resultIdDigestValueMap[v_counterIdDigestValueMap] := {
    v_eMAID.id,
    fx_convertOctetstring2Base64Binary(
    fx_messageDigest("SHA-256",
    fx_canonicalExi(eMAIDTypeAsBitstring))) };
} else {
    log("Given BodyType not relevant for Signature verification")
}
return resultIdDigestValueMap;
}

function f_verifySignature(in template (value) BodyType p_bodyType,
in template (value) SignatureType p_signatureType,
in template (value) X509CertificateType p_certificate,
in verdicttype v_vct)
return boolean {

var integer v_counter_for_referenceList;
var integer v_counter_for_idDigestValueMap;
var charstring currentIdInReferenceList;
var charstring currentIdInIdDigestValueMap;
var charstring currentDigestValueInReferenceList;
var charstring currentDigestValueInIdDigestValueMap;

// contains calculated digestValues for all ids in bodytype message
var idDigestValueMap calcIdDigestValueMap :=
f_calculateIdDigestMapFromBodyType(p_bodyType);
// contains digestValues and signatureValue from header
var SignedInfo signedInfo := p_signatureType.signedInfo;

//Digest comparison procedure:
// a) Iterate over elements in calculated id-digest-hashmap (-> outer loop)
// b) Iterate over elements in signedInfo (-> inner loop)
for (v_counter_for_idDigestValueMap := 0;
v_counter_for_idDigestValueMap < lengthof(calcIdDigestValueMap);
v_counter_for_idDigestValueMap := v_counter_for_idDigestValueMap +1) {

var boolean matchedDigestValue := false;
var boolean matchedId := false;

currentIdInIdDigestValueMap :=
calcIdDigestValueMap[v_counter_for_idDigestValueMap].id;
currentDigestValueInIdDigestValueMap :=
calcIdDigestValueMap[v_counter_for_idDigestValueMap].digestValue;

for (v_counter_for_referenceList := 0;

```

```

        v_counter_for_referenceList < lengthof(signedInfo.reference_list);
        v_counter_for_referenceList := v_counter_for_referenceList +1) {

    var ReferenceType refType :=
        signedInfo.reference_list[v_counter_for_referenceList];
    var charstring refTypeUri := refType.uRI;
    var charstring refTypeUriWithoutHashtag :=
        substr(refTypeUri, 1, (lengthof(refTypeUri)-1)); //remove #

    currentIdInReferenceList := refTypeUriWithoutHashtag;
    currentDigestValueInReferenceList := refType.digestValue;

    // 2-stepped comparison
    // -> first      check if IDs do exist
    // -> second     compare digestValue for matched ID
    if (currentIdInReferenceList == currentIdInIdDigestValueMap) {
        matchedId := true;

        if (currentDigestValueInReferenceList ==
            currentDigestValueInIdDigestValueMap) {
            //log("DigestValue MATCH: ", currentDigestValueInReferenceList);
            matchedDigestValue := true;
        } else {
            log("Calculated digestValue ",currentDigestValueInIdDigestValueMap,
                " does NOT match digestValue in SignedInfo ",
                currentDigestValueInReferenceList, " for Id ",
                currentIdInReferenceList);
        }
    }

    //NOTICE: Already return on first "ID mismatch"
    if (not matchedId) {
        setverdict(v_vct, "ID ", currentIdInIdDigestValueMap,
            " could NOT be found in SignedInfo element ",
            signedInfo);
        return false;
    }

    //NOTICE: Already return on first digestValue mismatch!!!
    if (not matchedDigestValue) {
        setverdict(v_vct, "Digest MISMATCH");
        return false;
    }
};

var charstring signatureCharstringValue := p_signatureType.
    signatureValue.base;
var bitstring encodedSignedInfoType :=encvalue(signedInfo);

var octetstring exiEncodedSignedInfoTypeOctetstring :=
    fx_canonicalExi(encodedSignedInfoType);

var boolean verified :=
    fx_verifySignedInfoDigest("SHA256withECDSA",
        exiEncodedSignedInfoTypeOctetstring,
        signatureCharstringValue,
        p_certificate.raw_Certificate
    );

if(verified) {
    setverdict(pass, "Signature verification was successful.");
}

return verified;
}

function f_calculateSalesTariffReferenceListFromChargeParameterDiscoveryRes(
    in template (value) BodyType p_BodyType) return LocalReferenceList {

    var integer v_counter;
    var LocalReferenceList list;

    var idDigestValueMap salesTariffIdDigestValueMap :=
        f_calculateIdDigestMapFromBodyType(p_BodyType);
    for (v_counter := 0; v_counter < lengthof(
        salesTariffIdDigestValueMap); v_counter:= v_counter +1) {

```

```
        list[v_counter] := md_ReferenceType_001(  
            salesTariffIdDigestValueMap[v_counter].id,  
            salesTariffIdDigestValueMap[v_counter].digestValue);  
    }  
    return list;  
}  
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO 15118-4:2018

Annex G (normative)

Function specifications for 15118-2 V2GTP

This annex covers all *test behaviors* for V2GTP in the ATS.

G.1 SECC functions

```

module TestBehavior_SECC_V2GTPSessionSetup {

    import from Timer_15118_2 all;
    import from ComponentsAndPorts all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Templates_CMN_SupportedAppProtocol all;
    import from Templates_SECC_ServiceDiscovery all;
    import from Templates_CMN_SessionSetup all;
    import from Templates_CMN_V2GMessage all;
    import from Templates_CMN_V2GTPMessage all;
    import from DataStructure_V2GTP all;

    function f_SECC_CMN_TB_VTB_V2GTPSessionSetup_001() runs on SECC_Tester return verdicttype {

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SessionSetupReq);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2GTP_Message_001(
            md_CMN_V2GTP_header_TYPE_001('01'H,'FE'H,'8001'H, omit),
            md_CMN_V2GTP_payload_TYPE_002(
                md_CMN_V2G_Message_001(
                    md_CMN_MessageHeaderType_001(vc_SessionID),
                    md_CMN_BodyType_001(
                        md_CMN_CMN_SessionSetupReq_001(vc_eVCCID))))));

        alt {
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2GTP_Message_001(
                md_CMN_V2GTP_header_TYPE_001('01'H,'FE'H,'8001'H,?),
                md_CMN_V2GTP_payload_TYPE_002(
                    md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(?),
                        md_CMN_BodyType_001(
                            md_CMN_CMN_SessionSetupRes_001(?,?,*))))));

                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"V2GTP (SessionSetupRes) message was correct.");
            }
            [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
                "terminated by the SUT.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received anything, but not " &
                "the expected V2GTP message.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_CommunicationSetup_Timeout(fail);
        }
        return getverdict;
    }

    function f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002(in template V2GTP_header_TYPE v_v2gtpHeader)
        runs on SECC_Tester return verdicttype {

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SessionSetupReq);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2GTP_Message_001(v_v2gtpHeader,
            md_CMN_V2GTP_payload_TYPE_002(
                md_CMN_V2G_Message_001(
                    md_CMN_MessageHeaderType_001(vc_SessionID),
                    md_CMN_BodyType_001(
                        md_CMN_CMN_SessionSetupReq_001(vc_eVCCID))))));

        alt {
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2GTP_Message_001(

```

```

md_CMN_V2GTP_header_TYPE_001(
  '01'H,'FE'H,'8001'H,?),
md_CMN_V2GTP_payload_TYPE_002(
md_CMN_V2G_Message_001(
md_CMN_MessageHeaderType_001(?),
md_CMN_BodyType_001(
md_CMN_CMN_SessionSetupRes_001(?,?,*)))) {

  tc_V2G_EVCC_Msg_Timer.stop;
  setverdict(fail,"Invalid V2GTP header was not ignored.");
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
  "terminated by the SUT.");
[] a_SECC_Unexpected_Message_Content(fail, "Received anything, but not the " &
  "expected V2GTP message.");
[] a_SECC_Timeout(pass) {
  setverdict(pass,"Invalid V2GTP header was ignored.");
}
[] a_SECC_CommunicationSetup_Timeout(fail);
}
return getverdict;
}
}

module TestBehavior_SECC_V2GTPSDP {

import from ComponentsAndPorts all;
import from Timer_15118_2 all;
import from Templates_CMN_SDP all;
import from Templates_CMN_V2GTPMessage all;
import from DataStructure_SDP all;
import from DataStructure_V2GTP all;
import from TestBehavior_SECC_CommonBehavior all;
import from Services_HAL_61851 all;

function f_SECC_CMN_TB_VTB_V2GTPSDP_001() runs on SECC_Tester return verdicttype {

  var V2GTP_Message v_responseMessage;
  var integer v_count := 0;

  while(v_count < vc_maxRepetitionSDP){

    tc_EVCC_SDP_Timer.start(par_EVCC_SDP_Timeout);
    pt_V2G_UDP_SDP_Port.send(md_CMN_V2GTP_Message_001(
      md_CMN_V2GTP_header_TYPE_001('01'H,'FE'H,'9000'H, omit),
      md_CMN_V2GTP_payload_TYPE_001(
        md_CMN_SDP_Request_TYPE_001('10'H,'00'H))));

    v_count := v_count + 1;

    alt {
      [] pt_V2G_UDP_SDP_Port.receive(md_CMN_V2GTP_Message_001(
        md_CMN_V2GTP_header_TYPE_001(
          '01'H,'FE'H,'9001'H,'00000014'H),
          md_CMN_V2GTP_payload_TYPE_001(
            md_CMN_SDP_Response_TYPE_001(?, ?, ?, ?)))
        -> value v_responseMessage {

          tc_EVCC_SDP_Timer.stop;
          vc_V2G_Port_IPAddress := v_responseMessage.v2gtp_payload.content.sdp_message.
            content.sdp_Response.secc_IPAddress;
          vc_V2G_Port_PortNumber := v_responseMessage.v2gtp_payload.content.sdp_message.
            content.sdp_Response.secc_Port;

          v_count := vc_maxRepetitionSDP;
          setverdict(pass,"V2GTP Header message was correct. " &
            "(SDP Response Message).");
        }
      [] pt_V2G_UDP_SDP_Port.receive {
        repeat;
      }
      [] tc_EVCC_SDP_Timer.timeout {
        // send a new SDP request message if v_count < vc_maxRepetitionSDP
        if(v_count == vc_maxRepetitionSDP){
          setverdict(fail,"SDP repetition maximum is reached. SDP process " &
            "shall be stopped.");
        }
      }
    }
  }
}
}

```

```

        [] a_SECC_CommunicationSetup_Timeout(fail);
    }
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_V2GTPSDP_002(in template V2GTP_header_TYPE v_v2gtpHeader)
    runs on SECC_Tester return verdicttype {

    var V2GTP_Message v_responseMessage;

    tc_EVCC_SDP_Timer.start(par_EVCC_SDP_Timeout);
    pt_V2G_UDP_SDP_Port.send(md_CMN_V2GTP_Message_001(v_v2gtpHeader,
        md_CMN_V2GTP_payload_TYPE_001(
            md_CMN_SDP_Request_TYPE_001('10'H,'00'H))));

    alt {
        [] pt_V2G_UDP_SDP_Port.receive(md_CMN_V2GTP_Message_001(
            md_CMN_V2GTP_header_TYPE_001(
                '01'H,'FE'H,'9001'H,'00000014'H),
            md_CMN_V2GTP_payload_TYPE_001(
                md_CMN_SDP_Response_TYPE_001(?, ?, ?, ?))))
            -> value v_responseMessage {

                tc_EVCC_SDP_Timer.stop;
                setverdict(fail,"Invalid V2GTP header was not ignored.");
            }
        [] pt_V2G_UDP_SDP_Port.receive {
            setverdict(fail,"Invalid V2GTP header was not ignored.");
        }
        [] tc_EVCC_SDP_Timer.timeout {
            setverdict(pass,"Invalid V2GTP header was ignored.");
        }
        [] a_SECC_CommunicationSetup_Timeout(fail);
    }
    return getverdict;
}
}

```

G.2 EVCC functions

```

module TestBehavior_EVCC_V2GTPSessionSetup {

    import from ComponentsAndPorts all;
    import from Timer_15118_2 all;
    import from Services_HAL_61851 all;
    import from Templates_CMN_SupportedAppProtocol all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from Templates_CMN_SessionSetup all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_CMN_V2GMessage all;
    import from Templates_CMN_V2GTPMessage all;
    import from urn_iso_15118_2_2010_AppProtocol language "XSD" all;
    import from TestBehavior_EVCC_ServiceDiscovery all;
    import from DataStructure_V2GTP all;
    import from Timer_15118 all;

    function f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_001() runs on EVCC_Tester return verdicttype {

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2GTP_Message_001(
            md_CMN_V2GTP_header_TYPE_001('01'H, 'FE'H, '8001'H, omit),
            md_CMN_V2GTP_payload_TYPE_004(
                md_CMN_CMN_SupportedAppProtocolRes_001(
                    oK_SuccessfulNegotiation,(vc_SchemaID))));

        alt
        {
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2GTP_Message_001(
                md_CMN_V2GTP_header_TYPE_001('01'H, 'FE'H,
                    '8001'H, ?),
                md_CMN_V2GTP_payload_TYPE_002(
                    md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(?),
                        md_CMN_BodyType_001(
                            md_CMN_CMN_SessionSetupReq_001(?))))))
        }
    }
}

```

```

    {
        tc_V2G_SECC_Sequence_Timer.stop;
        setverdict(pass, "V2GTP (SessionSetupReq) message was correct.");
    }
    [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
        "terminated by the SUT.");
    [] a_EVCC_Unexpected_Message_Content(fail, "Received anything, but not the " &
        "expected V2GTP message.");
    [] a_EVCC_Timeout(fail);
}
return getverdict;
}

function f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002(in template V2GTP_header_TYPE v_v2gtpHeader)
    runs on EVCC_Tester return verdicttype {

    vc_SessionID := f_rnd_SessionID(1,429496);
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

    // send Session Setup response message with invalid header information
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2GTP_Message_001(
        v_v2gtpHeader, md_CMN_V2GTP_payload_TYPE_002(
            md_CMN_V2G_Message_001(md_CMN_MessageHeaderType_001(
                vc_SessionID), md_CMN_BodyType_001(
                    md_CMN_CMN_SessionSetupRes_001(
                        oK_NewSessionEstablished, vc_EvseId, omit))))));

    timer t_wait;

    // Wait until SUT message timeout
    t_wait.start(par_V2G_EVCC_Msg_Timeout_SessionSetupReq -
        par_CMN_Transmission_Delay);
    alt {
        [] t_wait.timeout;
    }

    tc_CMN_TCP_Connection_Termination_Timer.start(par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
            "terminated by the SUT.") {
            setverdict(pass, "Session Setup response message with invalid header " &
                "information was ignored.");
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2GTP_Message_001(
            md_CMN_V2GTP_header_TYPE_001(?, ?, ?, ?),
            md_CMN_V2GTP_payload_TYPE_002(
                md_CMN_V2G_Message_001(
                    md_CMN_MessageHeaderType_001(?),
                    md_CMN_BodyType_001(
                        md_CMN_CMN_ServiceDiscoveryReq_003(*, *))))))
        {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail, "V2GTP (SessionSetupReq) message was not expected.");
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
            "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout();
        [] a_EVCC_Timeout(fail);
    }
    return getverdict;
}

}

module TestBehavior_EVCC_V2GTPSDP {

    import from ComponentsAndPorts all;
    import from Timer_15118_2 all;
    import from Templates_CMN_SDP all;
    import from Templates_CMN_V2GTPMessage all;
    import from DataStructure_SDP all;
    import from DataStructure_V2GTP all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Services_HAL_61851 all;
    import from Configurations_15118_2 all;

    function f_EVCC_CMN_TB_VTB_V2GTPSDP_001() runs on EVCC_Tester return verdicttype {

```

```

alt
{
    [] pt_V2G_UDP_SDP_Port.receive(md_CMN_V2GTP_Message_001(
        md_CMN_V2GTP_header_TYPE_001('01'H, 'FE'H,
            '9000'H, '00000002'H),
        md_CMN_V2GTP_payload_TYPE_001(
            md_CMN_SDP_Request_TYPE_001(?, ?)))
    {
        setverdict(pass, "V2GTP Header message was correct. (SDP Request Message).");
    }
    [] a_EVCC_No_SDP_Message("Received anything, but no SDP_Message", fail);
    [] a_EVCC_SDP_Timeout(fail);
}
return getverdict;
}

function f_EVCC_CMN_TB_VTB_V2GTPSDP_002(in template V2GTP_header_TYPE v_v2gtpHeader)
    runs on EVCC_Tester return verdicttype

var V2GTP_Message v_requestMessage;
var boolean v_repetition := true;

while(v_repetition){
    alt
    {
        [] pt_V2G_UDP_SDP_Port.receive(md_CMN_V2GTP_Message_001(
            md_CMN_V2GTP_header_TYPE_001('01'H, 'FE'H,
                '9000'H, '00000002'H),
            md_CMN_V2GTP_payload_TYPE_001(
                md_CMN_SDP_Request_TYPE_001(?, ?)))
            -> value v_requestMessage {

            vc_TransportProtocol := v_requestMessage.v2gtp_payload.content.
                sdp_message.content.sdp_Request.transportProtocol;
            vc_Security := v_requestMessage.v2gtp_payload.content.sdp_message.
                content.sdp_Request.security;

            if(not v_repetition){
                setverdict(pass, "SDP response message with invalid header " &
                    "information was ignored. " &
                    "EVCC has repeated SDP request message.");
                break;
            }

            // send SDP response message with invalid header information
            pt_V2G_UDP_SDP_Port.send(md_CMN_V2GTP_Message_001(
                v_v2gtpHeader, md_CMN_V2GTP_payload_TYPE_001(
                    md_CMN_SDP_Response_TYPE_001(
                        par SECC_IPAddress, vc_V2G_Port_PortNumber,
                        vc_Security, vc_TransportProtocol)))));

            v_repetition := false;
            repeat;
        }
        [] a_EVCC_No_SDP_Message("Received anything, but no SDP_Message", fail){
            v_repetition := false;
        }
        [] a_EVCC_SDP_Timeout(fail){
            v_repetition := false;
        }
    }
}
return getverdict;
}
}

```

Annex H (normative)

Function specifications for 15118-2 SDP messages

This annex covers all *test behaviors* for SDP in the ATS.

H.1 SECC functions

```

module TestBehavior_SECC_SDP {

    import from ComponentsAndPorts all;
    import from Timer_15118_2 all;
    import from Templates_CMN_SDP all;
    import from DataStructure_SDP all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Services_HAL_61851 all;

    function f_SECC_CMN_TB_VTB_SDP_001(in Security_TYPE v_security,
                                        in verdicttype v_vct)
        runs on SECC_Tester
        return verdicttype {

            var integer v_count := 0;
            var SDP_Message v_responseMessage;

            while(v_count < vc_maxRepetitionSDP){
                tc_EVCC_SDP_Timer.start(par_EVCC_SDP_Timeout);
                pt_V2G_UDP_SDP_Port.send(md_CMN_SDP_Request_TYPE_001(v_security,'00'H)) to cc_multicast;
                v_count := v_count + 1;

                alt {
                    [] pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Response_TYPE_001(
                        ?, (49152 .. 65535), v_security, '00'H))
                        -> value v_responseMessage {

                            tc_EVCC_SDP_Timer.stop;
                            vc_V2G_Port_IPAddress := v_responseMessage.content.sdp_Response.secc_IPAddress;
                            vc_V2G_Port_PortNumber := v_responseMessage.content.sdp_Response.secc_Port;
                            vc_Security := v_responseMessage.content.sdp_Response.security;
                            v_count := vc_maxRepetitionSDP;
                            setverdict(pass,"SDP Response message was correct.");
                        }

                    [] pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Response_TYPE_001(
                        ?, (49152 .. 65535), ?, '00'H))
                        -> value v_responseMessage {

                            tc_EVCC_SDP_Timer.stop;
                            v_count := vc_maxRepetitionSDP;
                            setverdict(inconc,"SUT did not support the requested transmission security.");
                        }

                    [] pt_V2G_UDP_SDP_Port.receive {
                        repeat;
                    }

                    [] tc_EVCC_SDP_Timer.timeout {
                        // send a new SDP request message if v_count < vc_maxRepetitionSDP
                        if(v_count == vc_maxRepetitionSDP){
                            setverdict(v_vct,"SDP repetition maximum is reached. SDP process " &
                                "shall be stopped.");
                        }
                    }

                    [] a_SECC_CommunicationSetup_Timeout(v_vct);
                }
            }

            return getverdict;
        }

    function f_SECC_CMN_TB_VTB_SDP_002(in Security_TYPE v_security) runs on SECC_Tester
        return verdicttype {

            alt {
                [] tc_V2G_SECC_CommunicationSetup_Timer.timeout;
            }
        }
    }
}

```

```

    [] pt_V2G_UDP_SDP_Port.receive {
        setverdict(fail, "No SDP message was expected.");
    }
}

tc_EVCC_SDP_Timer.start(par_EVCC_SDP_Timeout);
pt_V2G_UDP_SDP_Port.send(md_CMN_SDP_Request_TYPE_001(v_security,'00'H))
                        to cc_multicast;

alt {
    [] pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Response_TYPE_001(
        ?, (49152 .. 65535), ?, ?)){
        tc_EVCC_SDP_Timer.stop;
        setverdict(fail,"The SUT did not stop the SDP server.");
    }
    [] tc_EVCC_SDP_Timer.timeout {
        setverdict(pass,"No SDP response was received. The SUT has " &
            "stopped the SDP server.");
    }
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_SDP_003() runs on SECC_Tester return verdicttype {

    var integer v_count := 0;
    var SDP_Message v_responseMessage;

    while(v_count < vc_maxRepetitionSDP){
        tc_EVCC_SDP_Timer.start(par_EVCC_SDP_Timeout);
        pt_V2G_UDP_SDP_Port.send(md_CMN_SDP_Request_TYPE_001('00'H,'00'H)) to cc_multicast;
        v_count := v_count + 1;

        alt {
            [] pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Response_TYPE_001(
                ?, (49152 .. 65535), '01'H, '00'H))
                -> value v_responseMessage {

                tc_EVCC_SDP_Timer.stop;
                vc_V2G_Port_IPAddress := v_responseMessage.content.sdp_Response.secc_IPAddress;
                vc_V2G_Port_PortNumber := v_responseMessage.content.sdp_Response.secc_Port;
                vc_Security := v_responseMessage.content.sdp_Response.security;
                v_count := vc_maxRepetitionSDP;
                setverdict(pass,"SDP Response message was correct.");
            }
            [] pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Response_TYPE_001(
                ?, (49152 .. 65535), ?, '00'H))
                -> value v_responseMessage {

                tc_EVCC_SDP_Timer.stop;
                v_count := vc_maxRepetitionSDP;
                setverdict(inconc,"SUT supports the requested transmission security " &
                    "'secured with TLS'. Check the PIXIT configuration.");
            }
            [] pt_V2G_UDP_SDP_Port.receive {
                repeat;
            }
            [] tc_EVCC_SDP_Timer.timeout {
                // send a new SDP request message if v_count < vc_maxRepetitionSDP
                if(v_count == vc_maxRepetitionSDP){
                    setverdict(fail,"SDP repetition maximum is reached. SDP process " &
                        "shall be stopped.");
                }
            }
            [] a_SECC_CommunicationSetup_Timeout(fail);
        }
    }
    return getverdict;
}
}

```

H.2 EVCC functions

```

module TestBehavior_EVCC_SDP {

    import from ComponentsAndPorts all;
    import from Templates_CMN_SDP all;

```



```

import from DataStructure_SDP all;
import from Timer_15118_2 all;
import from TestBehavior_EVCC_CommonBehavior all;
import from Templates_EVCC_SupportedAppProtocol all;
import from Configurations_15118_2 all;
import from Services_Security all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from Templates_CMN_Security all;
import from urn_iso_15118_2_2013_X509Def all;
import from Pics_15118_2 all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from DataStructure_V2G_TCP_TLS_Port_Control all;

function f_EVCC_CMN_TB_VTB_SDP_001(in template Security_TYPE v_security,
                                   in verdicttype v_vct)
    runs on EVCC_Tester return verdicttype {

    var SDP_Message v_requestMessage;

    alt {

        []pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Request_TYPE_001(v_security, '00'H))
            -> value v_requestMessage
            sender vc_sut_ipAddress.ipAddress {

            vc_TransportProtocol := v_requestMessage.content.sdp_Request.transportProtocol;
            vc_Security := v_requestMessage.content.sdp_Request.security;
            tc_V2G_SECC_CommunicationSetup_Timer.stop;
            setverdict(pass,"SDP Request message was correct.");
        }
        []pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Request_TYPE_001(?, '00'H)) {

            tc_V2G_SECC_CommunicationSetup_Timer.stop;
            setverdict(inconc,"SUT did not request the correct transmission security.");
        }
        [] a_EVCC_No_SDP_Message("Received anything, but no SDP_Message", v_vct);
        [] a_EVCC_SDP_Timeout(v_vct);
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_SDP_002() runs on EVCC_Tester return verdicttype {

    var SDP_Message v_requestMessage;
    var integer v_count := 0;

    alt {
        []pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Request_TYPE_001(?, '00'H))
            -> value v_requestMessage {

            v_count := v_count + 1;
            log("Number of received SDP messages: ", v_count);
            repeat;
        }
        [] a_EVCC_No_SDP_Message("Received anything, but no SDP_Message", fail);
        [v_count == vc_maxRepetitionSDP] tc_V2G_SECC_CommunicationSetup_Timer.timeout {
            setverdict(pass,"The repetition limit of (vc_maxRepetitionSDP -1) " &
                "is reached. EVCC has stopped the SECC Discovery.");
        }
        [v_count < vc_maxRepetitionSDP or v_count > vc_maxRepetitionSDP]
            tc_V2G_SECC_CommunicationSetup_Timer.timeout {
            setverdict(fail,"EVCC has stopped the SECC Discovery without " &
                "consideration of the repetition limit");
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_SDP_003() runs on EVCC_Tester return verdicttype {

    var SDP_Message v_requestMessage;
    var integer v_count := 0;
    var SDP_Message v_requestMessageSDP;
    var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_responseMessage;
    var V2G_TCP_TLS_Port_Control_Ack_TYPE ackType;

    alt {
        []pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Request_TYPE_001(?, '00'H))
            -> value v_requestMessage
    }

```

```

sender vc_sut_ipAddress.ipAddress {

vc_TransportProtocol := v_requestMessage.content.sdp_Request.transportProtocol;
vc_Security := v_requestMessage.content.sdp_Request.security;
if(vc_Security == cc_hexTcp) {
    vc_v2g_tcp_tls_parameter := md_CMN_CMN_TcpParameter_001(vc_sut_ipAddress.ipAddress,
                                                            vc_V2G_Port_PortNumber,
                                                            vc_Security);
} else if(vc_Security == cc_hexTls) {

    vc_v2g_tcp_tls_parameter := md_CMN_CMN_TlsParameter_001(vc_sut_ipAddress.ipAddress,
                                                            vc_V2G_Port_PortNumber,
                                                            vc_Security, vc_cipherSuite,
                                                            vc_enabledProtocol,
                                                            md_CertificateChainType_002(
                                                            vc_leafCert, vc_subCert2,
                                                            vc_subCert1));

} else {log("Invalid security type was chosen.");};

if(getverdict == pass) {
    f_EVCC_CMN_PR_InitConfiguration_002(system, fail);

    tc_EVCC_SDP_Timer.start(par_EVCC_SDP_Timeout);
    v_count := v_count + 1;
    tc_EVCC_SDP_Timer.timeout;
    log("Number of received SDP messages: ", v_count);

    if(v_count > 1) {

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

        pt_V2G_UDP_SDP_Port.send(md_CMN_SDP_Response_TYPE_001(
                                par_SECC_IPAddress, vc_V2G_Port_PortNumber,
                                vc_Security, vc_TransportProtocol));

        alt {
            //SDP Handling if more than one message is received from SUT
            []pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Request_TYPE_001(?, '00'H))
                -> value v_requestMessageSDP {
                pt_V2G_UDP_SDP_Port.send(md_CMN_SDP_Response_TYPE_001(
                    par_SECC_IPAddress,
                    vc_V2G_Port_PortNumber,
                    vc_Security, vc_TransportProtocol));

                repeat;
            }
            []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_CMN_tcpTlsStatusRes_001(
                e_connectPort,?,?))
                -> value v_responseMessage {
                var boolean isClosed := v_responseMessage.result.portStatus.closed;
                ackType := v_responseMessage.ack;
                if(ackType == e_success) {
                    setverdict(pass, "TCP or TLS handshake was successful.");
                } else if(ackType == e_failed) {
                    setverdict(fail, "TCP or TLS handshake was not successful.");
                }
                repeat;
            }
            []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                mw_EVCC_CMN_SupportedAppProtocolReq_001()) {
                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(pass, "SupportedAppProtocolReq message was correct. " &
                    "The SDP client is able to handle " &
                    "asynchronously arriving SECC Discovery " &
                    "Response messages");
            }
            [isError()] a_EVCC_Unexpected_Message(fail, "V2G_Message received, but " &
                "SupportedAppProtocolReq " &
                "was expected.");
            [ a_EVCC_Unexpected_Message_Content(fail, "Received " &
                "SupportedAppProtocolReq " &
                "message, but with invalid " &
                "data content.");

            [ a_EVCC_Timeout(fail);

        }
    }
} else {
    repeat;
}

```

```

    }
  }
  [] a_EVCC_No_SDP_Message("Received anything, but no SDP_Message", fail);
  [] a_EVCC_SDP_Timeout(fail);
}
return getverdict;
}

function f_EVCC_CMN_TB_VTB_SDP_004() runs on EVCC_Tester return verdicttype {
  var SDP_Message v_requestMessage;

  alt {
    []pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Request_TYPE_001(?,'00'H))
      -> value v_requestMessage {

      vc_TransportProtocol := v_requestMessage.content.sdp_Request.transportProtocol;
      vc_Security := v_requestMessage.content.sdp_Request.security;

      pt_V2G_UDP_SDP_Port.send(md_CMN_SDP_Request_TYPE_001(vc_Security, vc_TransportProtocol));

      alt {
        []pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Request_TYPE_001(?,'00'H)) {
          setverdict(pass,"The SDP client did not reply to SDP Request message.");
        }
        [] pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Response_TYPE_001(?,(49152 .. 65535),
          vc_Security, '00'H)) {
          setverdict(fail,"The SDP client did reply to SDP Request message.");
        }
        [] a_EVCC_No_SDP_Message("Received anything, but no SDP_Message", fail);
        [] a_EVCC_SDP_Timeout(fail);
      }
    }
    [] a_EVCC_No_SDP_Message("Received anything, but no SDP_Message", fail);
    [] a_EVCC_SDP_Timeout(fail);
  }
  return getverdict;
}
}

```

Annex I (normative)

Function specifications for 15118-2 V2G application layer messages

This annex covers all *test behaviors* for the V2G application layer messages in the ATS.

I.1 SECC functions

I.1.1 V2G protocol handshake

I.1.1.1 SECC functions for SupportedAppProtocol

```

module TestBehavior_SECC_SupportedAppProtocol {

    import from Services_HAL_61851 all;
    import from urn_iso_15118_2_2010_AppProtocol all;
    import from Configurations_15118_2 all;
    import from ComponentsAndPorts all;
    import from Timer_15118_2 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Templates_CMN_SupportedAppProtocol all;
    import from Templates_SECC_SupportedAppProtocol all;
    import from Templates_CMN_SessionSetup all;
    import from Templates_CMN_V2GMessage all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Services_HAL_61851 all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Pixit_15118_2 all;
    import from Timer_15118 all;

    function f_SECC_CMN_TB_VTB_SupportedAppProtocol_001(in verdicttype v_vct)
        runs on SECC_Tester
        return verdicttype {

        var SupportedAppProtocolRes v_responseMessage;
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SupportedAppProtocolReq);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_SupportedAppProtocolReq_001(
            md_CMN_CMN_AppProtocolType_001(
                vc_Supported_ISO_Namespace,
                vc_versionNumberMajor,
                vc_versionNumberMinor,10,2)));

        alt {
            [pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_CMN_SupportedAppProtocolRes_001(
                m_SECC_CMN_validResponseCodes(),*))
                -> value v_responseMessage {
                tc_V2G_EVCC_Msg_Timer.stop;
                if(v_responseMessage.responseCode == oK_SuccessfulNegotiation){
                    setverdict(pass,"SupportedAppProtocolRes message was correct.");
                }
                else {
                    setverdict(pass,"SupportedAppProtocolRes message was correct. " &
                        "Minor version deviation was detected.");
                }
            }
            [ a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
                "terminated by the SUT.");
            [isError()] a_SECC_Unexpected_Message(v_vct, "V2G_Message received, but " &
                "SupportedAppProtocolRes was expected.");
            [ a_SECC_Unexpected_Message_Content(v_vct, "Received SupportedAppProtocolRes message, "
                "but with invalid data content.");
            [ a_SECC_Timeout(v_vct);
            [ a_SECC_CommunicationSetup_Timeout(v_vct);
        }
        return getverdict;
    }
}

```

```

}

function f_SECC_CMN_TB_VTB_SupportedAppProtocol_002(in verdicttype v_vct)
    runs on SECC_Tester
    return verdicttype {

    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SupportedAppProtocolReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_SupportedAppProtocolReq_001(
        md_CMN_CMN_AppProtocolType_001(
            "urn:iso:15118:2:2014:MsgDef",3,
            vc_versionNumberMinor,1,1)));

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_CMN_SupportedAppProtocolRes_002(
            failed_NoNegotiation)) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"SupportedAppProtocolRes message was correct. " &
                "Protocol handshake failed");

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                e_getPortStatus)) to "local";
            tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
            alt{
                [isShutdownOSC == false]
                a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                    isShutdownOSC := true;
                    repeat;
                }
                [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                    "terminated by the SUT.") {
                    if(isShutdownOSC == false) {
                        logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                    pt_HAL_61851_Internal_Port.clear;
                }
                [] a_SECC_Unexpected_Message_Content(v_vct, "Received any message, " &
                    "but TCP connection was " &
                    ">>NOT<< terminated by the SUT");
                [] a_SECC_TCP_Connection_Termination_Timeout();
            }
        }
        [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
            "terminated by the SUT.");
        [isError()] a_SECC_Unexpected_Message(v_vct, "V2G Message received, but " &
            "SupportedAppRes was expected.");
        [] a_SECC_Unexpected_Message_Content(v_vct, "Received SupportedAppRes message, " &
            "but with invalid data content.");
        [] a_SECC_Timeout(v_vct);
        [] a_SECC_CommunicationSetup_Timeout(v_vct);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_SupportedAppProtocol_003() runs on SECC_Tester return verdicttype {

    timer t_wait;
    var boolean isShutdownOSC := false;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout -
        par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_getPortStatus)) to "local";

    alt {

```

```

    [] t_wait.timeout;
}

f_SECC_setIsConfirmationFlagDC();
f_SECC_changeValidFrequencyRange(0,0);
f_SECC_changeValidDutyCycleRange(100,100);
deactivate(vc_Default_IEC_61851_ListenerBehavior);

tc_CMN_TCP_Connection_Termination_Timer.start(
par_CMN_TCP_Connection_Termination_Timeout);

alt {
    []
    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
    }
    [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
        "terminated by the SUT.")
        if(isShutdownOSC == false) {
            logWarn("SECC oscillator was not shut down before the " &
                "TCP connection was terminated.");
            pt_HAL_61851_Internal_Port.clear;
        }
    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
        "but TCP connection was " &
        ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_SupportedAppProtocol_004() runs on SECC_Tester return verdicttype {
    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SupportedAppProtocolReq);

    vc_versionNumberMinor := PIXIT_SECC_CMN_VersionNumberMinor + 1;
    if(vc_versionNumberMinor < vc_versionNumberMajor) {

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_SupportedAppProtocolReq_001(
            md_CMN_CMN_AppProtocolType_001(
                vc_Supported_ISO_Namespace,
                vc_versionNumberMajor,
                vc_versionNumberMinor,10,2)));

        alt {
            [pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_CMN_SupportedAppProtocolRes_001(
                oK_SuccessfulNegotiationWithMinorDeviation,*)) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"SupportedAppProtocolRes message was correct. " &
                    "Minor version deviation was detected.");
            }
            [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
                "terminated by the SUT.");
            [isError()] a_SECC_Unexpected_Message(fail, "V2G Message received, but " &
                "SupportedAppProtocolRes " &
                "was expected.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received SupportedAppProtocolRes " &
                "message, but with invalid data " &
                "content.");
            [a_SECC_Timeout(fail);
            a_SECC_CommunicationSetup_Timeout(fail);
            ]
        }
    } else {
        setverdict(inconc, "The selected VersionNumberMinor is not smaller " &
            "than VersionNumberMajor.");
    }
    return getverdict;
}
}

```

I.1.2 V2G messages

I.1.2.1 SECC functions for SessionSetup

```

module TestBehavior_SECC_SessionSetup {

    import from Timer_15118_2 all;
    import from ComponentsAndPorts {type SECC_Tester;;};
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Templates_SECC_ServiceDiscovery all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_CMN_SessionSetup all;
    import from Templates_CMN_V2GMessage all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Services_HAL_61851 all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from TestBehavior_EVCC_ServiceDiscovery all;
    import from LibFunctions_15118_2 all;
    import from Timer_15118 all;

    function f_is_SessionID_not_zero(SessionIDType sessionID) return boolean {

        if(oct2int(sessionID)==0){
            return false;
        }
        else {
            return true;
        }
        return true;
    }

    function f_SECC_CMN_TB_VTB_SessionSetup_001(in verdicttype v_vct)
        runs on SECC_Tester
        return verdicttype {

            var V2G_Message v_responseMessage;
            var charstring v_message;

            tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SessionSetupReq);

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_001(vc_SessionID),
                md_CMN_BodyType_001(md_CMN_CMN_SessionSetupReq_001(
                    vc_eVCCID))));

            alt {
                [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                    md_CMN_MessageHeaderType_001(?),
                    md_CMN_BodyType_001(
                        md_CMN_CMN_SessionSetupRes_001(
                            oK_NewSessionEstablished,?,*)))
                    -> value v_responseMessage {

                        tc_V2G_EVCC_Msg_Timer.stop;
                        vc_SessionID := v_responseMessage.header.sessionID;
                        vc_EvseId := v_responseMessage.body.bodyElement.sessionSetupRes.eVSEID;
                        log("EVSEID: " & toString(charstring>(vc_EvseId));
                        if(vc_EvseId != cc_EvseId_NotSup) {
                            var boolean result := fx_validateEVSEID(vc_EvseId, "ISO");
                            if(not result) {
                                setverdict(v_vct,"Invalid EVSEID syntax was detected.");
                            }
                        }
                        if(getverdict == pass) {
                            if(not f_is_SessionID_not_zero(vc_SessionID)){
                                setverdict(v_vct,"Invalid SessionID was detected.");
                            }
                        }
                        else{
                            setverdict(pass,"SessionSetup message was correct.");
                        }
                    }
                }
            }

            [] a_SECC_TCPCConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
                "terminated by the SUT.");
            [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                {sessionSetupRes := ?});
            [isError()] a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes received, " &
                "but V2G_Message was expected.");
            [] a_SECC_Unexpected_Message_Content(v_vct, "Received SessionSetupRes message, " &
                "but with invalid data content.");
        }
    }
}

```



```

        [] a_SECC_Timeout(v_vct);
        [] a_SECC_CommunicationSetup_Timeout(v_vct);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_SessionSetup_002() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SessionSetupReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_CMN_CMN_ServiceDiscoveryReq_002(omit)),0);

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(?),
            md_CMN_BodyType_001(
                mdw_SECC_CMN_ServiceDiscoveryRes_001(
                    FAILED_SequenceError)))) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                "'FAILED_SequenceError' is successful.");

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                e_getPortStatus)) to "local";
            tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
            alt{
                [isShutdownOSC == false]
                a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                    isShutdownOSC := true;
                    repeat;
                }
                [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                    "terminated by the SUT.") {
                    if(isShutdownOSC == false) {
                        logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");
                        pt_HAL_61851_Internal_Port.clear;
                    }
                }
                [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                    "but TCP connection was " &
                    ">>NOT<< terminated by the SUT");
                [] a_SECC_TCP_Connection_Termination_Timeout();
            }
            [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
                "terminated by the SUT.");
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                {serviceDiscoveryRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
                "but V2G_Message was expected.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received SessionSetupRes message, " &
                "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_CommunicationSetup_Timeout(fail);
        }
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_SessionSetup_003() runs on SECC_Tester return verdicttype {

    timer t_wait;
    var boolean isShutdownOSC := false;

```

```

// Wait until sequence timeout in SUT should expire
t_wait.start(par_V2G_SECC_Sequence_Timeout -
             par_CMN_Transmission_Delay);
pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                e_getPortStatus)) to "local";

alt {
    [] t_wait.timeout;
}

f_SECC_setIsConfirmationFlagDC();
f_SECC_changeValidFrequencyRange(0,0);
f_SECC_changeValidDutyCycleRange(100,100);
deactivate(vc_Default_IEC_61851_ListenerBehavior);

tc_CMN_TCP_Connection_Termination_Timer.start(
par_CMN_TCP_Connection_Termination_Timeout);

alt {
    []
    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
    }
    [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                           "terminated by the SUT.") {
        if(isShutdownOSC == false) {
            logWarn("SECC oscillator was not shut down before the " &
                  "TCP connection was terminated.");
            pt_HAL_61851_Internal_Port.clear;
        }
    }
    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                           "but TCP connection was " &
                                           ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_SessionSetup_004() runs on SECC_Tester return verdicttype {

    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    // Wait until V2G_SECC_CommunicationSetup timeout in SUT should expire
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus)) to "local";

    alt {
        [] tc_V2G_SECC_CommunicationSetup_Timer.timeout;
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
            tc_V2G_SECC_CommunicationSetup_Timer.stop;
            setverdict(fail, "No V2G message was expected.");
        }
    }

    tc_CMN_TCP_Connection_Termination_Timer.start(
par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                  "terminated by the SUT.") {
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                      "TCP connection was terminated.");
                pt_HAL_61851_Internal_Port.clear;
            }
        }
        [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &

```

```

                                "but TCP connection was " &
                                ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_SessionSetup_005(in verdicttype v_vct)
    runs on SECC_Tester
    return verdicttype {

    var V2G_Message v_responseMessage;
    var charstring v_message;

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SessionSetupReq);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_CMN_CMN_SessionSetupReq_001(vc_eVCCID)));

    alt {
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_CMN_CMN_SessionSetupRes_001(
                    oK_OldSessionJoined,?,*))))
            -> value v_responseMessage {

            tc_V2G_EVCC_Msg_Timer.stop;
            vc_EvseId := v_responseMessage.body.bodyElement.sessionSetupRes.evseId;
            log("EVSEID: " & toString<charstring>(vc_EvseId));
            if(vc_EvseId != cc_EvseId_NotSup) {
                var boolean result := fx_validateEVSEID(vc_EvseId, "ISO");
                if(not result) {
                    setverdict(v_vct,"Invalid EVSEID syntax was detected.");
                }
            }
            if(getverdict == pass) {
                setverdict(pass,"SessionSetupRes message with responseCode " &
                    "'oK_OldSessionJoined' was correct after " &
                    "'paused session.'");
            }
        }

        [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
            "terminated by the SUT.");
        [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
            {sessionSetupRes := ?});
        [isError() | a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes received, " &
            "but V2G Message was expected.");
        [] a_SECC_Unexpected_Message_Content(v_vct, "Received SessionSetupRes message, " &
            "but with invalid data content.");
        [] a_SECC_Timeout(v_vct);
        [] a_SECC_CommunicationSetup_Timeout(v_vct);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_SessionSetup_006(in boolean v_isZero) runs on SECC_Tester
    return verdicttype {

    var V2G_Message v_responseMessage;
    var charstring v_message;
    var SessionIDType v_SessionID_old;

    v_SessionID_old := vc_SessionID;
    if(v_isZero) {
        vc_SessionID := '00000000000000000';
    }
    else {
        vc_SessionID := f_rnd_SessionID(1,429496);
    }

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SessionSetupReq);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),

```

```

md_CMN_BodyType_001(
md_CMN_CMN_SessionSetupReq_001(vc_eVCCID))););

alt {
[]    pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(?),
        md_CMN_BodyType_001(
        md_CMN_CMN_SessionSetupRes_001(
        oK_NewSessionEstablished,?,*)))
        -> value v_responseMessage {

tc_V2G_EVCC_Msg_Timer.stop;
vc_SessionID := v_responseMessage.header.sessionID;
vc_EvseId := v_responseMessage.body.bodyElement.sessionSetupRes.eVSEID;
log("EVSEID: " & toString<charstring>(vc_EvseId));
if(vc_EvseId != cc_EvseId_NotSup) {
var boolean result := fx_validateEVSEID(vc_EvseId, "ISO");
    if(not result) {
        setverdict(fail,"Invalid EVSEID syntax was detected.");
    }
}
if(getverdict == pass) {
    if(f_is_SessionID_not_zero(vc_SessionID) and
    (vc_SessionID != v_SessionID_old)){
        setverdict(pass,"SessionSetupRes message with responseCode " &
        "'oK_NewSessionEstablished' was correct " &
        "after paused session.");
    } else {
        setverdict(fail,"The received SessionID is not unequal " &
        "to '0' and not unequal to the SessionID value "
        "stored from the preceding V2G Communication "
        "Session.");
    }
}
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {sessionSetupRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received SessionSetupRes message, " &
    "but with invalid data content.");
[] a_SECC_Timeout(fail);
[] a_SECC_CommunicationSetup_Timeout(fail);
}
return getverdict;
}
}

```

I.1.2.2 SECC functions for ServiceDiscovery

```

module TestBehavior_SECC_ServiceDiscovery {

import from Services_HAL_61851 all;
import from Timer_15118_2 all;
import from Pics_15118 all;
import from Pics_15118_2 all;
import from ComponentsAndPorts all;
import from Templates_CMN_ServiceDiscovery all;
import from Templates_SECC_ServiceDiscovery all;
import from Templates_CMN_SessionSetup all;
import from Templates_CMN_Authorization all;
import from Templates_CMN_SupportedAppProtocol all;
import from Templates_CMN_V2GMessage all;
import from TestBehavior_SECC_CommonBehavior all;
import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
import from TTlibrary_Logging all;
import from TestBehavior_EVCC_ServiceDiscovery all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from Timer_15118 all;

function f_SECC_CMN_TB_VTB_ServiceDiscovery_001(in verdicttype v_vct)
    runs on SECC_Tester
    return verdicttype {

var V2G_Message v_responseMessage;

```

```

var charstring v_message;

tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);

pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(vc_SessionID),
    md_CMN_BodyType_001(
        md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

alt {
    []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_SECC_CMN_ServiceDiscoveryRes_002(oK, ?, ?, *)))
        -> value v_responseMessage{

        tc_V2G_EVCC_Msg_Timer.stop;
        vc_supportedEnergyTransferMode := v_responseMessage.body.bodyElement.
            serviceDiscoveryRes.chargeService.
            supportedEnergyTransferMode;

        vc_paymentOptionList := v_responseMessage.body.bodyElement.
            serviceDiscoveryRes.paymentOptionList;

        var boolean v_isSelected := false;
        for (var integer i:=0; i<sizeof(vc_paymentOptionList.paymentOption_list); i:=i + 1) {
            if(PICS_CMN_CMN_IdentificationMode == pnC) {
                if(vc_paymentOptionList.paymentOption_list[i] == contract) {
                    vc_SelectedPaymentOption := vc_paymentOptionList.paymentOption_list[i];
                    v_isSelected := true;
                }
            }
            else {
                if(vc_paymentOptionList.paymentOption_list[i] == externalPayment) {
                    vc_SelectedPaymentOption := vc_paymentOptionList.paymentOption_list[i];
                    v_isSelected := true;
                }
            }
        }
        if(not v_isSelected){
            setverdict(v_vct,"Invalid payment option was offered by the SUT.");
        }
        else{
            vc_serviceList := v_responseMessage.body.bodyElement.
                serviceDiscoveryRes.serviceList;
            vc_chargeService := v_responseMessage.body.bodyElement.
                serviceDiscoveryRes.chargeService;
            setverdict(pass,"ServiceDiscoveryRes message was correct.");
        }
    }
    [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
        "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
        {serviceDiscoveryRes := ?});
    [isError()] a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes received, " &
        "but V2G_Message was expected.");
    [] a_SECC_Unexpected_Message_Content(v_vct, "Received ServiceDiscoveryRes message, " &
        "but with invalid data content.");
    [] a_SECC_Timeout(v_vct);
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_ServiceDiscovery_002() runs on SECC_Tester return verdictttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(m_CMN_CMN_AuthorizationReq_001)));

    alt {

```

```

[]
a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
    isShutdownOSC := true;
    repeat;
}
[]pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(vc_SessionID),
    md_CMN_BodyType_001(
        md_CMN_CMN_AuthorizationRes_001(
            fAILED_SequenceError, ?)))) {

    tc_V2G_EVCC_Msg_Timer.stop;
    setverdict(pass, "AuthorizationRes with responseCode " &
        "'fAILED_SequenceError' is successful.");

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_getPortStatus)) to "local";
    tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);
    alt{
        [isShutdownOSC == false]
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
            "terminated by the SUT.") {
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                    "TCP connection was terminated.");
                pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                "but TCP connection was " &
                ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout();
        }
    }
    [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
        "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
        {authorizationRes := ?});
    [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
        "but V2G Message was expected.");
    [] a_SECC_Unexpected_Message_Content(fail, "Received ServiceDiscoveryRes message, " &
        "but with invalid data content.");
    [] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_ServiceDiscovery_003() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(
            f_rnd_SessionID(1,429496)),
        md_CMN_BodyType_001(
            m_CMN_CMN_ServiceDiscoveryReq_001)));

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(?),
            md_CMN_BodyType_001(
                md_SECC_CMN_ServiceDiscoveryRes_002(

```

```

        fFAILED_UnknownSession, ?,?,omit)))) {
    tc_V2G_EVCC_Msg_Timer.stop;
    setverdict(pass,"ServiceDiscoveryRes with responseCode " &
        "'fFAILED_UnknownSession' is successful.");

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_getPortStatus)) to "local";
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);
    alt{
        [isShutdownOSC == false]
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
            "terminated by the SUT.") {
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                    "TCP connection was terminated.");
                pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                "but TCP connection was " &
                ">NOT< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout();
        }
    }
    [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
        "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
        {serviceDiscoveryRes := ?});
    [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
        "but V2G Message was expected.");
    [] a_SECC_Unexpected_Message_Content(fail, "Received ServiceDiscoveryRes message, " &
        "but with invalid data content.");
    [] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_ServiceDiscovery_004() runs on SECC_Tester return verdicttype {
    timer t_wait;
    var boolean isShutdownOSC := false;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout -
        par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_getPortStatus)) to "local";

    alt {
        [] t_wait.timeout;
    }

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
            "terminated by the SUT.") {
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                    "TCP connection was terminated.");
                pt_HAL_61851_Internal_Port.clear;
            }
        }
        [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
            "but TCP connection was " &

```



```

        ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_ServiceDiscovery_005(in verdicttype v_vct)
    runs on SECC_Tester
    return verdicttype {

    var V2G_Message v_responseMessage;
    var charstring v_message;

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_CMN_CMN_ServiceDiscoveryReq_002(omit))););

    alt {
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_SECC_CMN_ServiceDiscoveryRes_002(
                    oK, ?,vc_chargeService,*))))
            -> value v_responseMessage;

        tc_V2G_EVCC_Msg_Timer.stop;
        vc_supportedEnergyTransferMode := v_responseMessage.body.bodyElement.
            serviceDiscoveryRes.chargeService.
            supportedEnergyTransferMode;

        vc_SelectedPaymentOption := v_responseMessage.body.bodyElement.serviceDiscoveryRes.
            paymentOptionList.paymentOption_list[0];
        if(PICS_CMN_CMN_IdentificationMode == pnC) {
            if(vc_SelectedPaymentOption != contract) {
                setverdict(v_vct, "Invalid payment option was chosen by the SUT.");
            }
        }
        else {
            if(vc_SelectedPaymentOption != externalPayment) {
                setverdict(v_vct, "Invalid payment option was chosen by the SUT.");
            }
        }
        vc_serviceList := v_responseMessage.body.bodyElement.serviceDiscoveryRes.
            serviceList;
        setverdict(pass,"ServiceDiscoveryRes message was correct.");
    }

    [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
        "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
        {serviceDiscoveryRes := ?});
    [isError()] a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes received, " &
        "but V2G_Message was expected.");
    [] a_SECC_Unexpected_Message_Content(v_vct, "Received ServiceDiscoveryRes message, " &
        "but with invalid data content.");
    [] a_SECC_Timeout(v_vct);
}
return getverdict;
}
}

```

I.1.2.3 SECC functions for ServiceDetail

```

module TestBehavior_SECC_ServiceDetail {

    import from Timer_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from ComponentsAndPorts all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Templates_SECC_ServiceDetail all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_SECC_ServiceDiscovery all;
    import from Templates_CMN_V2GMessage all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from TTlibrary_Logging all;

```

```

import from Pics_15118 all;
import from Services_HAL_61851 all;
import from TestBehavior_EVCC_ServiceDiscovery all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from TTlibrary_Logging all;
import from Pixit_15118_2 all;
import from Timer_15118 all;

function f_SECC_CMN_TB_VTB_ServiceDetail_001(in verdicttype v_vct)
    runs on SECC_Tester
    return verdicttype {

var integer counter := 0;
var charstring v_message;
var V2G_Message v_responseMessage;
var boolean v_result := true;

if(ispresent(vc_serviceList)) {
    if(PIXIT_SECC_CMN_InternetAccess == internetAccess) {
        v_result := f_checkServiceCategory(internet, vc_serviceList);
    }
    if(v_result) {
        while (counter < sizeof(vc_serviceList.service_list)) {
            var ServiceType v_service := vc_serviceList.service_list[counter].
                serviceType;
            f_checkServiceType(v_service, v_vct);
            if(getverdict == pass) {
                tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDetailReq);

                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                    md_CMN_MessageHeaderType_001(
                        vc_SessionID),
                    md_CMN_BodyType_001(
                        md_SECC_CMN_ServiceDetailReq_001(
                            v_service.serviceID))));

                alt {
                    []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(
                            vc_SessionID),
                        md_CMN_BodyType_001(
                            mdw_SECC_CMN_ServiceDetailRes_001(
                                v_service.serviceID))))
                        -> value v_responseMessage {

                        tc_V2G_EVCC_Msg_Timer.stop;
                        var ServiceParameterListType v_serviceParameterList :=
                            v_responseMessage.body.bodyElement.serviceDetailRes.
                                serviceParameterList;
                        f_checkServiceParameterList(v_serviceParameterList,
                            v_service, v_vct);

                        if(getverdict == pass) {
                            setverdict(pass, "ServiceDetailRes message was correct.");
                        }
                        counter := counter + 1;
                    }
                    [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                        "misleadingly " &
                        "terminated by " &
                        "the SUT.") {

                        counter := sizeof(vc_serviceList) + 1;
                        break;
                    }
                }
                [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message " &
                    "was received.",
                    {serviceDetailRes := ?}) {

                    counter := sizeof(vc_serviceList) + 1;
                    break;
                }
                [isError()] a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes " &
                    "received, but " &
                    "V2G_Message " &
                    "was expected.") {

                    counter := sizeof(vc_serviceList) + 1;
                    break;
                }
                [] a_SECC_Unexpected_Message_Content(v_vct, "Received ServiceDetailRes "

```

&

```

&
"message, but with invalid "
"data content.") {
    counter := sizeof(vc_serviceList) + 1;
    break;
}
[] a_SECC_Timeout(v_vct) {
    counter := sizeof(vc_serviceList) + 1;
    break;
}
}
else {
    counter := sizeof(vc_serviceList) + 1;
}
}
}
else {setverdict(inconc,"A serviceList is not available.");}
return getverdict;
}

function f_SECC_CMN_TB_VTB_ServiceDetail_002() runs on SECC_Tester return verdicttype {
    var integer counter := 0;
    var charstring v_message;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    if(ispresent(vc_serviceList)) {
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDetailReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_CMN_CMN_ServiceDiscoveryReq_002(omit))));
        alt {
            []
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_001(vc_SessionID),
                md_CMN_BodyType_001(
                    mdw_SECC_CMN_ServiceDiscoveryRes_001(
                        FAILED_SequenceError)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"ServiceDetailRes with responseCode " &
                    "'FAILED_SequenceError' is successful.");
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                    e_getPortStatus)) to "local";
                tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                alt{
                    [isShutdownOSC == false]
                    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                        isShutdownOSC := true;
                        repeat;
                    }
                    [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                        "terminated by the SUT.") {
                        if(isShutdownOSC == false) {
                            logWarn("SECC oscillator was not shut down before the " &
                                "TCP connection was terminated.");}
                        pt_HAL_61851_Internal_Port.clear;
                    }
                    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                        "but TCP connection was " &
                        ">>NOT<< terminated by the SUT");
                    [] a_SECC_TCP_Connection_Termination_Timeout();
                }
            }
            [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
                "terminated by the SUT.");
        }
    }
}

```

```

    [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                      {serviceDiscoveryRes := ?});
    [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
                                          "but V2G_Message was expected.");
    [] a_SECC_Unexpected_Message_Content(fail, "Received ServiceDetailRes message, " &
                                          "but with invalid data content.");
    [] a_SECC_Timeout(fail);
  }
}
else {setverdict(inconc, "A serviceList is not available.");}
return getverdict;
}

function f_SECC_CMN_TB_VTB_ServiceDetail_003() runs on SECC_Tester return verdicttype {

  var integer counter := 0;
  var charstring v_message;
  var boolean isShutdownOSC := false;

  f_SECC_setIsConfirmationFlagDC();
  f_SECC_changeValidFrequencyRange(0,0);
  f_SECC_changeValidDutyCycleRange(100,100);
  deactivate(vc_Default_IEC_61851_ListenerBehavior);

  if(ispresent(vc_serviceList)) {
    tc_V2G_EVCC_Msg_Timer.start(par V2G_EVCC_Msg_Timeout_ServiceDetailReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
      md_CMN_MessageHeaderType_001(
        f_rnd_SessionID(1,429496)),
      md_CMN_BodyType_001(
        md_SECC_CMN_ServiceDetailReq_001(
          vc_serviceList.service_list[0].serviceType.serviceID))));

    alt {
      []
      a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
      }
      []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(?),
        md_CMN_BodyType_001(
          mdw_SECC_CMN_ServiceDetailRes_003(
            fAILED_UnknownSession)))) {

        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(pass, "ServiceDetailRes with responseCode " &
                      "'fAILED_UnknownSession' is successful.");

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
          e_getPortStatus)) to "local";
        tc_CMN_TCP_Connection_Termination_Timer.start(
          par CMN_TCP_Connection_Termination_Timeout);
        alt{
          [isShutdownOSC == false]
          a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
          }
          [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                  "terminated by the SUT.") {

            if(isShutdownOSC == false) {
              logWarn("SECC oscillator was not shut down before the " &
                    "TCP connection was terminated.");
              pt_HAL_61851_Internal_Port.clear;
            }
          }
          [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                  "but TCP connection was " &
                                                  ">>NOT<< terminated by the SUT");
          [] a_SECC_TCP_Connection_Termination_Timeout();
        }
      }
      [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
                                                  "terminated by the SUT.");
      [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                      {serviceDetailRes := ?});
      [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
                                          "but V2G_Message was expected.");
      [] a_SECC_Unexpected_Message_Content(fail, "Received ServiceDetailRes message, " &
                                          "but with invalid data content.");
    }
  }
}

```

```

        [] a_SECC_Timeout(fail);
    }
}
else {setverdict(inconc,"A serviceList is not available.");}
return getverdict;
}

function f_SECC_CMN_TB_VTB_ServiceDetail_004() runs on SECC_Tester return verdicttype {

    var integer counter := 0;
    var charstring v_message;
    var ServiceIDType v_invalidServiceID := 65535;
    var boolean isInvalid;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    if(ispresent(vc_serviceList)) {
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDetailReq);
        isInvalid := f_checkInvalidServiceID(v_invalidServiceID, vc_serviceList);
        if(isInvalid) {
            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_001(vc_SessionID),
                md_CMN_BodyType_001(
                    md_SECC_CMN_ServiceDetailReq_001(
                        v_invalidServiceID))));
        }

        alt {
            []
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_001(vc_SessionID),
                md_CMN_BodyType_001(
                    mdw_SECC_CMN_ServiceDetailRes_003(
                        FAILED_ServiceIDInvalid)))) {

                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"ServiceDetailRes with responseCode " &
                    "'FAILED_ServiceIDInvalid' is successful.");

                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                    e_getPortStatus)) to "local";
                tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                alt{
                    [isShutdownOSC == false]
                    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                        isShutdownOSC := true;
                        repeat;
                    }
                    [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                        "terminated by the SUT.") {
                        if(isShutdownOSC == false) {
                            logWarn("SECC oscillator was not shut down before the " &
                                "TCP connection was terminated.");}
                        pt_HAL_61851_Internal_Port.clear;
                    }
                    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                        "but TCP connection was " &
                        ">>NOT<< terminated " &
                        "by the SUT");
                    [] a_SECC_TCP_Connection_Termination_Timeout();
                }
            }
            [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                "misleadingly " &
                "terminated by the SUT.");
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                {serviceDetailRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes " &
                "received, but V2G_Message " &
                "was expected.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received ServiceDetailRes " &
                "message, but with invalid " &

```

```

        "data content.");
        [] a_SECC_Timeout(fail);
    }
}
else {setverdict(inconc,"A serviceList is not available.");}
return getverdict;
}

function f_SECC_CMN_TB_VTB_ServiceDetail_005() runs on SECC_Tester return verdicttype {

    timer t_wait;
    var boolean isShutdownOSC := false;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout -
        par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_getPortStatus)) to "local";

    alt {
        [] t_wait.timeout;
    }

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
            "terminated by the SUT.") {
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                    "TCP connection was terminated.");}
            pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
            "but TCP connection was " &
            ">>NOT<< terminated by the SUT");
        [] a_SECC_TCP_Connection_Termination_Timeout();
    }
    return getverdict;
}

function f_checkInvalidServiceID(ServiceIDType v_invalidServiceID,
    ServiceListType vc_serviceList) return boolean {
    for (var integer i:=0; i<sizeof(vc_serviceList); i:=i + 1) {
        if(v_invalidServiceID == vc_serviceList.service_list[0].serviceType.serviceID){
            setverdict(inconc,"Selected serviceID is not invalid.");
            return false;
        }
    }
    return true;
}

function f_checkServiceType(ServiceType v_service, in verdicttype v_vct) {
    if(v_service.serviceID == 1) {
        setverdict(v_vct, "Service Id '1' is reserved for the service " &
            "category 'EVcharging'.");
    }
    else if(v_service.serviceID == 2) {
        if(v_service.serviceCategory != contractCertificate) {
            setverdict(v_vct, "Invalid service category was used.");
        }
    }
    else if(v_service.serviceID == 3) {
        if(v_service.serviceCategory != internet) {
            setverdict(v_vct, "Invalid service category was used.");
        }
    }
}

```

```

    }
}
else if(v_service.serviceID == 4) {
    if(v_service.serviceCategory != otherCustom) {
        setverdict(v_vct, "Invalid service category was used.");
    }
}
else {
    log("Unknown service type was used.");
    log("Service ID: " & toString<integer>(v_service.serviceID));
    log("Service Name: " & toString<charstring>(v_service.serviceName));
    log("Service Category: " &
        toString<ServiceCategoryType>(v_service.serviceCategory));
}
}

function f_checkServiceParameterList(ServiceParameterListType v_serviceParameterList,
    ServiceType v_service, in verdicttype v_vct) {

    if(v_service.serviceID == 2) {
        for (var integer i:=0; i<sizeof(v_serviceParameterList); i:=i + 1) {
            if(v_serviceParameterList.parameterSet_list[i].parameterSetID == 1) {
                if(v_serviceParameterList.parameterSet_list[i].
                    parameter_list[0].name != "Service") {
                    setverdict(v_vct, "Invalid ParameterName (Service) was used for the " &
                        "parameter serviceID equals to 2 and " &
                        "'parameterSetID' equals to 1.");

                    break;
                }
            }
            else if (v_serviceParameterList.parameterSet_list[i].parameter_list[0].
                choice.stringValue != "Installation") {
                setverdict(v_vct, "Invalid stringValue was used for the parameter " &
                    "serviceID equals to 2 and " &
                    "'parameterSetID' equals to 1.");
            }
        }
        else if(v_serviceParameterList.parameterSet_list[i].parameterSetID == 2) {
            if(v_serviceParameterList.parameterSet_list[i].
                parameter_list[0].name != "Service") {
                setverdict(v_vct, "Invalid ParameterName (Service) was used for the " &
                    "parameter serviceID equals to 2 and " &
                    "'parameterSetID' equals to 2.");

                break;
            }
            else if (v_serviceParameterList.parameterSet_list[i].parameter_list[0].
                choice.stringValue != "Update") {
                setverdict(v_vct, "Invalid stringValue was used for the parameter " &
                    "serviceID equals to 2 and " &
                    "'parameterSetID' equals to 2.");
            }
        }
    }
}

if(v_service.serviceID == 3) {
    for (var integer i:=0; i<sizeof(v_serviceParameterList); i:=i + 1) {
        if(v_serviceParameterList.parameterSet_list[i].parameterSetID == 1) {
            if(v_serviceParameterList.parameterSet_list[i].
                parameter_list[0].name != "Protocol") {
                setverdict(v_vct, "Invalid ParameterName (Protocol) was used for the " &
                    "parameter serviceID equals to 3 and " &
                    "'parameterSetID' equals to 1.");

                break;
            }
            else if (v_serviceParameterList.parameterSet_list[i].parameter_list[0].
                choice.stringValue != "ftp") {
                setverdict(v_vct, "Invalid stringValue was used for the parameter " &
                    "serviceID equals to 3 and " &
                    "'parameterSetID' equals to 1.");

                break;
            }
            else if(v_serviceParameterList.parameterSet_list[i].
                parameter_list[1].name != "Port") {
                setverdict(v_vct, "Invalid ParameterName (Port) was used for the " &
                    "parameter serviceID equals to 3 and " &
                    "'parameterSetID' equals to 1.");

                break;
            }
        }
    }
}
}

```



```

else if (v_serviceParameterList.parameterSet_list[i].parameter_list[1].
choice.intValue != 20) {
    setverdict(v_vct, "Invalid intValue was used for the parameter " &
        "serviceID equals to 3 and " &
        "'parameterSetID' equals to 1.");
}
}
}
if(v_serviceParameterList.parameterSet_list[i].parameterSetID == 2) {
    if(v_serviceParameterList.parameterSet_list[i].
parameter_list[0].name != "Protocol") {
        setverdict(v_vct, "Invalid ParameterName (Protocol) was used for the " &
            "parameter serviceID equals to 3 and " &
            "'parameterSetID' equals to 1.");
        break;
    }
    else if (v_serviceParameterList.parameterSet_list[i].parameter_list[0].
choice.stringValue != "ftp") {
        setverdict(v_vct, "Invalid stringValue was used for the parameter " &
            "serviceID equals to 3 and " &
            "'parameterSetID' equals to 1.");
        break;
    }
    else if(v_serviceParameterList.parameterSet_list[i].
parameter_list[1].name != "Port") {
        setverdict(v_vct, "Invalid ParameterName (Port) was used for the " &
            "parameter serviceID equals to 3 and " &
            "'parameterSetID' equals to 1.");
        break;
    }
    else if (v_serviceParameterList.parameterSet_list[i].parameter_list[1].
choice.intValue != 21) {
        setverdict(v_vct, "Invalid intValue was used for the parameter " &
            "serviceID equals to 3 and " &
            "'parameterSetID' equals to 1.");
    }
}
}
if(v_serviceParameterList.parameterSet_list[i].parameterSetID == 3) {
    if(v_serviceParameterList.parameterSet_list[i].
parameter_list[0].name != "Protocol") {
        setverdict(v_vct, "Invalid ParameterName (Protocol) was used for the " &
            "parameter serviceID equals to 3 and " &
            "'parameterSetID' equals to 1.");
        break;
    }
    else if (v_serviceParameterList.parameterSet_list[i].parameter_list[0].
choice.stringValue != "http") {
        setverdict(v_vct, "Invalid stringValue was used for the parameter " &
            "serviceID equals to 3 and " &
            "'parameterSetID' equals to 1.");
        break;
    }
    else if(v_serviceParameterList.parameterSet_list[i].
parameter_list[1].name != "Port") {
        setverdict(v_vct, "Invalid ParameterName (Port) was used for the " &
            "parameter serviceID equals to 3 and " &
            "'parameterSetID' equals to 1.");
        break;
    }
    else if (v_serviceParameterList.parameterSet_list[i].parameter_list[1].
choice.intValue != 80) {
        setverdict(v_vct, "Invalid intValue was used for the parameter " &
            "serviceID equals to 3 and " &
            "'parameterSetID' equals to 1.");
    }
}
}
if(v_serviceParameterList.parameterSet_list[i].parameterSetID == 4) {
    if(v_serviceParameterList.parameterSet_list[i].
parameter_list[0].name != "Protocol") {
        setverdict(v_vct, "Invalid ParameterName (Protocol) was used for the " &
            "parameter serviceID equals to 3 and " &
            "'parameterSetID' equals to 1.");
        break;
    }
    else if (v_serviceParameterList.parameterSet_list[i].parameter_list[0].
choice.stringValue != "https") {
        setverdict(v_vct, "Invalid stringValue was used for the parameter " &

```

```

        "serviceID equals to 3 and " &
        "'parameterSetID' equals to 1.");
    break;
}
else if(v_serviceParameterList.parameterSet_list[i].
parameter_list[1].name != "Port") {
    setverdict(v_vct, "Invalid ParameterName (Port) was used for the " &
        "parameter serviceID equals to 3 and " &
        "'parameterSetID' equals to 1.");
    break;
}
else if (v_serviceParameterList.parameterSet_list[i].parameter_list[1].
choice.intValue != 443) {
    setverdict(v_vct, "Invalid intValue was used for the parameter " &
        "serviceID equals to 3 and " &
        "'parameterSetID' equals to 1.");
}
}
}
}
}

function f_checkServiceCategory(ServiceCategoryType v_serviceCategory,
    ServiceListType vc_serviceList) return boolean {

    for (var integer i:=0; i<sizeof(vc_serviceList); i:=i + 1) {
        if(v_serviceCategory == vc_serviceList.service_list[i].serviceType.serviceCategory){
            return true;
        }
    }
    setverdict(inconc,"Selected serviceCategory could not be found within serviceList.");
    return false;
}
}

```

I.1.2.4 SECC functions for PaymentServiceSelection

```

module TestBehavior_SECC_PaymentServiceSelection {

    import from Pics_15118_2 all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Services_HAL_61851 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from XSDAUX language "TCN-3:2013" all;
    import from Timer_15118_2 all;
    import from Templates_SECC_ServiceDiscovery all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from TestBehavior_SECC_ServiceDetail all;
    import from Templates_SECC_PaymentServiceSelection all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_CMN_V2GMessage all;
    import from ComponentsAndPorts all;
    import from TestBehavior_EVCC_ServiceDiscovery all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Timer_15118 all;

    function f_SECC_CMN_TB_VTB_PaymentServiceSelection_001(in verdicttype v_vct)
        runs on SECC_Tester
        return verdicttype {

        var PaymentOptionType selPaymentOption;
        var SessionIDType sessionID := vc_SessionID;
        var charstring strMsg;
        var V2G_Message v_responseMessage;
        var charstring v_message;

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq);

        if (PICS_CMN_CMN_IdentificationMode == pnc){
            selPaymentOption := contract;
        }
        else{
            selPaymentOption := externalPayment;
        }
    }
}

```

```

pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(sessionID),
    md_CMN_BodyType_001(
        md_SECC_CMN_PaymentServiceSelectionReq_001(
            selPaymentOption,1)))));

alt {
    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001 (
            md_SECC_CMN_PaymentServiceSelectionRes_001(oK))))
        -> value v_responseMessage {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict ( pass,"PaymentServiceSelectionRes message was correct.");
        }
    [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
        "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
        {paymentServiceSelectionRes := ?});
    [isError()] a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes received, " &
        "but V2G Message was expected.");
    [] a_SECC_Unexpected_Message_Content(v_vct, "Received PaymentServiceSelectionRes message, "
        "but with invalid data content.");

    [] a_SECC_Timeout(v_vct);
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentServiceSelection_002() runs on SECC_Tester
    return verdicttype {

var PaymentOptionType selPaymentOption;
var SessionIDType sessionID := vc_SessionID;
var charstring strMsg;
var charstring v_message;
var boolean isShutdownOSC := false;

if ( PICS_CMN_CMN_IdentificationMode == pnC ){
    selPaymentOption := contract;
}
else {
    selPaymentOption := externalPayment;
}

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

    alt
    {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                mdw_SECC_CMN_ServiceDiscoveryRes_001(
                    FAILED_SequenceError)))) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                "'FAILED_SequenceError' is successful.");

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                e_getPortStatus)) to "local";
            tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
            alt{
                [isShutdownOSC == false]

```

```

a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
    isShutdownOSC := true;
    repeat;
}
[] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
    "terminated by the SUT.") {
    if(isShutdownOSC == false) {
        logWarn("SECC oscillator was not shut down before the " &
            "TCP connection was terminated.");}
    pt_HAL_61851_Internal_Port.clear;
}
[] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
    "but TCP connection was " &
    ">>NOT<< terminated by the SUT");
[] a_SECC_TCP_Connection_Termination_Timeout();
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received",
    {serviceDiscoveryRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionRes " &
    "message, but with invalid " &
    "data content.");
[] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentServiceSelection_003() runs on SECC_Tester
    return verdicttype {

var PaymentOptionType selPaymentOption;
var SessionIDType sessionID := vc_SessionID;
var charstring strMsg;
var charstring v_message;
var boolean isShutdownOSC := false;

if ( PICS_CMN_CMN_IdentificationMode == pnc ){
    selPaymentOption := contract;
}
else {
    selPaymentOption := externalPayment;
}

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq);
pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(
        f_rnd_SessionID(1,429496)),
    md_CMN_BodyType_001 (
        md_SECC_CMN_PaymentServiceSelectionReq_001(
            selPaymentOption,1))));
alt{
[]
a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
    isShutdownOSC := true;
    repeat;
}
[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(?),
    md_CMN_BodyType_001 (
        md_SECC_CMN_PaymentServiceSelectionRes_001(
            fAILED_UnknownSession)))) {

    tc_V2G_EVCC_Msg_Timer.stop;
    setverdict(pass, "PaymentServiceSelectionRes with responseCode " &
        "'fAILED_UnknownSession' is successful.");

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_getPortStatus)) to "local";
    tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);
    alt{

```

```

[isShutdownOSC == false]
a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
    isShutdownOSC := true;
    repeat;
}
[] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
    "terminated by the SUT.") {
    if(isShutdownOSC == false) {
        logWarn("SECC oscillator was not shut down before the " &
            "TCP connection was terminated.");
        pt_HAL_61851_Internal_Port.clear;
    }
    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
        "but TCP connection was " &
        ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {paymentServiceSelectionRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionRes " &
    "message, but with invalid " &
    "data content.");

[] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentServiceSelection_004() runs on SECC_Tester
    return verdicttype {

var PaymentOptionType selPaymentOption;
var charstring v_message;
var ServiceIDType v_invalidServiceID := 65535;
var boolean isInvalid;
var boolean isShutdownOSC := false;

if ( PICS_CMN_CMN_IdentificationMode == pnC ) {
    selPaymentOption := contract;
}
else {
    selPaymentOption := externalPayment;
}

isInvalid := f_checkInvalidServiceID(v_invalidServiceID, vc_serviceList);

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

if(isInvalid) {
    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_SECC_CMN_PaymentServiceSelectionReq_002(
                selPaymentOption,v_invalidServiceID,omit)))));

alt
{
[]
a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
    isShutdownOSC := true;
    repeat;
}

[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(vc_SessionID),
    md_CMN_BodyType_001 (
        md_SECC_CMN_PaymentServiceSelectionRes_001(
            FAILED_ServiceSelectionInvalid)))) {

    tc_V2G_EVCC_Msg_Timer.stop;
    setverdict(pass,"PaymentServiceSelectionRes with responseCode " &
        "'FAILED_ServiceSelectionInvalid' is successful.");

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(

```

```

        e_getPortStatus)) to "local";
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);
    alt{
        [isShutdownOSC == false]
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
            "terminated by the SUT.") {
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                    "TCP connection was terminated.");}
            pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
            "but TCP connection was " &
            ">>NOT<< terminated by the SUT");
        [] a_SECC_TCP_Connection_Termination_Timeout();
    }
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {paymentServiceSelectionRes :- ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionRes " &
    "message, but with invalid " &
    "data content.");
[] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentServiceSelection_005() runs on SECC_Tester
    return verdictttype {

    var PaymentOptionType selPaymentOption;
    var charstring v_message;
    var boolean isShutdownOSC := false;

    if (PICS_CMN_CMN_IdentificationMode == pnC) {
        selPaymentOption := contract;
    }
    else{
        selPaymentOption := externalPayment;
    }
    vc_ServiceId := 2;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001 (
            md_SECC_CMN_PaymentServiceSelectionReq_002(
                selPaymentOption,vc_ServiceId,omit))));

    alt
    {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_SECC_CMN_PaymentServiceSelectionRes_001(
                    FAILED_NoChargeServiceSelected)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"PaymentServiceSelectionRes with responseCode " &
                "'FAILED_NoChargeServiceSelected' is successful.");
        }
    }
}

```

```

pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
    e_getPortStatus)) to "local";
tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);
alt{
    [isShutdownOSC == false]
    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
    }
    [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
        "terminated by the SUT.") {
        if(isShutdownOSC == false) {
            logWarn("SECC oscillator was not shut down before the " &
                "TCP connection was terminated.");}
            pt_HAL_61851_Internal_Port.clear;
        }
    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
        "but TCP connection was " &
        ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {paymentServiceSelectionRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionRes " &
    "message, but with invalid " &
    "data content.");

[] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentServiceSelection_006() runs on SECC_Tester
    return verdicttype {

    timer t_wait;
    var boolean isShutdownOSC := false;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout -
        par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_getPortStatus)) to "local";

    alt {
        [] t_wait.timeout;
    }

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
            "terminated by the SUT.") {
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                    "TCP connection was terminated.");}
                pt_HAL_61851_Internal_Port.clear;
            }
        [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
            "but TCP connection was " &
            ">>NOT<< terminated by the SUT");
        [] a_SECC_TCP_Connection_Termination_Timeout();
    }
}

```



```

    return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentServiceSelection_007() runs on SECC_Tester
    return verdicttype {

var PaymentOptionType selPaymentOption;
var charstring v_message;
var boolean isShutdownOSC := false;

if(sizeof(vc_paymentOptionList.paymentOption_list) == 1) {
    if(PICS_CMN_CMN_IdentificationMode == pnC) {
        if(vc_paymentOptionList.paymentOption_list[0] != contract) {
            setverdict(inconc, "Invalid payment option was selected.");
        } else {
            selPaymentOption := externalPayment;
        }
    }
    else {
        if(vc_paymentOptionList.paymentOption_list[0] != externalPayment) {
            setverdict(inconc, "Invalid payment option was selected.");
        } else {
            selPaymentOption := contract;
        }
    }
} else {
    setverdict(inconc, "The SUT has provided more than one payment option.");
}

f_SECC_setIsConfirmationFlagDC();
f_SECC_changeValidFrequencyRange(0,0);
f_SECC_changeValidDutyCycleRange(100,100);
deactivate(vc_Default_IEC_61851_ListenerBehavior);

if(getverdict == pass) {

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(
            vc_SessionID),
        md_CMN_BodyType_001(
            md_SECC_CMN_PaymentServiceSelectionReq_001(
                selPaymentOption,1))));

    alt
    {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_SECC_CMN_PaymentServiceSelectionRes_001(
                    fAILED_PaymentSelectionInvalid)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"PaymentServiceSelectionRes with responseCode " &
                "'fAILED_PaymentSelectionInvalid' is successful.");

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                e_getPortStatus)) to "local";
            tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
            alt{
                [isShutdownOSC == false]
                a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                    isShutdownOSC := true;
                    repeat;
                }
                [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                    "terminated by the SUT.") {
                    if(isShutdownOSC == false) {
                        logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                    pt_HAL_61851_Internal_Port.clear;
                }
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                "but TCP connection was " &
                ">>NOT<< terminated by the SUT");
    }
}

```

```

        [] a_SECC_TCP_Connection_Termination_Timeout();
    }
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {paymentServiceSelectionRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received_PaymentServiceSelectionRes " &
    "message, but with invalid " &
    "data content.");

[] a_SECC_Timeout(fail);
}
}
return getverdict;
}
}
}

```

I.1.2.5 SECC functions for PaymentDetails

```

module TestBehavior_SECC_PaymentDetails {

    import from Timer_15118_2 all;
    import from Pics_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from ComponentsAndPorts all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Templates_SECC_PaymentDetails all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_SECC_ServiceDiscovery all;
    import from Templates_CMN_V2GMessage all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from TtLibrary_Logging all;
    import from Pics_15118 all;
    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_ServiceDiscovery all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Templates_CMN_Security all;
    import from Services_Security all;
    import from LibFunctions_15118_2 all;
    import from urn_iso_15118_2_2013_X509Def language "XSD" all;
    import from Timer_15118 all;

    function f_SECC_CMN_TB_VTB_PaymentDetails_001(in verdicttype v_vct)
        runs on SECC_Tester
        return verdicttype {

        var charstring v_message;
        var V2G_Message v_responseMessage;

        if(not(isbound(vc_contractSignatureCertChain))) {
            var X509CertificateType v_leafCert :=
                fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_CONTRACT_LEAF_VALID");
            var X509CertificateType v_subCert2 :=
                fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB2_VALID");
            var X509CertificateType v_subCert1 :=
                fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB1_VALID");
            vc_contractSignatureCertChain := md_CertificateChainType_002(v_leafCert,
                v_subCert2, v_subCert1)
        }

        if(getverdict == pass) {
            var charstring vc_EMAID := fx_getEMAID(vc_contractSignatureCertChain.
                x509Certificate);

            tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentDetailsReq);
            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_001(vc_SessionID),
                md_CMN_BodyType_001(
                    md_SECC_CMN_PaymentDetailsReq_001(vc_EMAID,
                        vc_contractSignatureCertChain))));

            alt {
                [pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                    md_CMN_MessageHeaderType_001(vc_SessionID),
                    md_CMN_BodyType_001(
                        md_SECC_CMN_PaymentDetailsRes_001(?,?)))))
            }
        }
    }
}

```

```

-> value v_responseMessage {

    vc_GenChallenge := v_responseMessage.body.bodyElement.
        paymentDetailsRes.genChallenge;
    tc_V2G_EVCC_Msg_Timer.stop;
    setverdict(pass, "PaymentDetailsRes message was correct.");
}
[] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
    {paymentDetailsRes := ?});
[isError()] a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(v_vct, "Received PaymentDetailsRes message, " &
    "but with invalid data content.");
[] a_SECC_Timeout(v_vct);
}
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentDetails_002() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentDetailsReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

    alt {
    []
    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
    }
    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            mdw_SECC_CMN_ServiceDiscoveryRes_001(
                fAILED_SequenceError)))) {

        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(pass, "ServiceDiscoveryRes with responseCode " &
            "'fAILED_SequenceError' is successful.");

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
            e_getPortStatus)) to "local";
        tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
        alt {
        [isShutdownOSC == false]
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
            "terminated by the SUT.") {
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                    "TCP connection was terminated.");}
            pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
            "but TCP connection was " &
            ">>NOT<< terminated by the SUT");

        [] a_SECC_TCP_Connection_Termination_Timeout();
    }
    }
    [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
        "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",

```

```

        {serviceDiscoveryRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received PaymentDetailsRes message, " &
    "but with invalid data content.");
[] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentDetails_003() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    if(not(isbound(vc_contractSignatureCertChain))) {
        var X509CertificateType v_leafCert :=
            fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_CONTRACT_LEAF_VALID");
        var X509CertificateType v_subCert2 :=
            fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB2_VALID");
        var X509CertificateType v_subCert1 :=
            fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB1_VALID");
        vc_contractSignatureCertChain := md_CertificateChainType_002(v_leafCert,
            v_subCert2, v_subCert1)
    }
    if(getverdict == pass) {
        var charstring vc_EMAID := fx_getEMAID(vc_contractSignatureCertChain.
            x509Certificate);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentDetailsReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(
                f_rnd_SessionID(1,429496)),
            md_CMN_BodyType_001(
                md_SECC_CMN_PaymentDetailsReq_001(vc_EMAID,
                    vc_contractSignatureCertChain))));

        alt {
            []
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_001(?),
                md_CMN_BodyType_001(
                    mdw_SECC_CMN_PaymentDetailsRes_002(
                        FAILED_UnknownSession)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass, "PaymentDetailsRes with responseCode " &
                    "'FAILED_UnknownSession' is successful.");

                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                    e_getPortStatus)) to "local";

                tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                alt{
                    [isShutdownOSC == false]
                    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                        isShutdownOSC := true;
                        repeat;
                    }
                    [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                        "terminated by the SUT.") {
                        if(isShutdownOSC == false) {
                            logWarn("SECC oscillator was not shut down before the " &
                                "TCP connection was terminated.");
                        }
                        pt_HAL_61851_Internal_Port.clear;
                    }
                    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                        "but TCP connection was " &
                        ">>NOT<< terminated by the SUT");
                    [] a_SECC_TCP_Connection_Termination_Timeout();
                }
            }
        }
    }
}

```

```

    }
  }
  [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.");
  [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {paymentDetailsRes := ?});
  [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.");
  [] a_SECC_Unexpected_Message_Content(fail, "Received PaymentDetailsRes message, " &
    "but with invalid data content.");
  [] a_SECC_Timeout(fail);
}
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentDetails_004() runs on SECC_Tester return verdicttype {
  timer t_wait;
  var boolean isShutdownOSC := false;

  // Wait until sequence timeout in SUT should expire
  t_wait.start(par_V2G_SECC_Sequence_Timeout -
    par_CMN_Transmission_Delay);
  pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
    e_getPortStatus)) to "local";

  alt {
    [] t_wait.timeout;
  }

  f_SECC_setIsConfirmationFlagDC();
  f_SECC_changeValidFrequencyRange(0,0);
  f_SECC_changeValidDutyCycleRange(100,100);
  deactivate(vc_Default_IEC_61851_ListenerBehavior);

  tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

  alt {
    []
    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
      isShutdownOSC := true;
      repeat;
    }
    [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
      "terminated by the SUT.") {
      if(isShutdownOSC == false) {
        logWarn("SECC oscillator was not shut down before the " &
          "TCP connection was terminated.");
        pt_HAL_61851_Internal_Port.clear;
      }
    }
    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
      "but TCP connection was " &
      ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
  }
  return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentDetails_005(in charstring v_certName,
  in charstring v_sub2CertName,
  in charstring v_sub1CertName,
  in template ResponseCodeType
  v_responseCode)
  runs on SECC_Tester
  return verdicttype {

  var charstring v_message;
  var boolean isShutdownOSC := false;
  var ResponseCodeType v_responseCodeReceived;
  var V2G_Message v_responseMessage;

  f_SECC_setIsConfirmationFlagDC();
  f_SECC_changeValidFrequencyRange(0,0);
  f_SECC_changeValidDutyCycleRange(100,100);
  deactivate(vc_Default_IEC_61851_ListenerBehavior);

  var X509CertificateType v_leafCert :=

```

```

fx_loadCertificate(PICS_CMN_CMN_PkiName, v_certName);
var X509CertificateType v_subCert2 :=
fx_loadCertificate(PICS_CMN_CMN_PkiName, v_sub2CertName);
var X509CertificateType v_subCert1 :=
fx_loadCertificate(PICS_CMN_CMN_PkiName, v_sub1CertName);
if(getverdict == pass) {
    var charstring vc_EMAID := fx_getEMAID(v_leafCert);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentDetailsReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_SECC_CMN_PaymentDetailsReq_001(vc_EMAID,
            md_CertificateChainType_002(v_leafCert,
            v_subCert2, v_subCert1))));

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(
                vc_SessionID),
            md_CMN_BodyType_001(
                mdw_SECC_CMN_PaymentDetailsRes_002(
                    v_responseCode)))
            -> value v_responseMessage {

            tc_V2G_EVCC_Msg_Timer.stop;
            v_responseCodeReceived := v_responseMessage.body.bodyElement.
                paymentDetailsRes.responseCode;
            setverdict(pass, "PaymentDetailsRes with responseCode '" &
                toString<ResponseCodeType>(v_responseCodeReceived) &
                "' is successful");

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                e_getPortStatus)) to "local";

            tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
            alt{
                [isShutdownOSC = false]
                a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                    isShutdownOSC := true;
                    repeat;
                }
                [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                    "terminated by the SUT.") {
                    if(isShutdownOSC == false) {
                        logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");
                        pt_HAL_61851_Internal_Port.clear;
                    }
                }
                [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                    "but TCP connection was " &
                    ">>NOT<< terminated by the SUT");
                [] a_SECC_TCP_Connection_Termination_Timeout();
            }
        }
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
            "terminated by the SUT.");
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
            {paymentDetailsRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
            "but V2G Message was expected.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received PaymentDetailsRes message, " &
            "but with invalid data content.");
        [] a_SECC_Timeout(fail);
    }
}
return getverdict;
}

```

I.1.2.6 SECC functions for Authorization

```
module TestBehavior_SECC_Authorization {
```

```

import from Timer_15118_2 all;
import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
import from urn_iso_15118_2_2013_MsgDef all;
import from LibFunctions_15118_2 all;
import from TestBehavior_SECC_CommonBehavior all;
import from Services_HAL_61851 all;
import from ComponentsAndPorts {type SECC_Tester;};
import from Templates_CMN_Authorization all;
import from Templates_CMN_ServiceDiscovery all;
import from Templates_CMN_V2GMessage all;
import from Templates_SECC_ServiceDiscovery all;
import from Pics_15118_2 all;
import from TTlibrary_Logging all;
import from Pics_15118 all;
import from TestBehavior_EVCC_ServiceDiscovery all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from Templates_CMN_Security all;
import from http_www_w3_org_2000_09_xmldsig all;
import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
import from Services_Security all;
import from urn_iso_15118_2_2013_X509Def all;
import from Pixit_15118_2 all;
import from Services_EIMIdentification all;
import from Timer_15118 all;

// EIM
function f_SECC_CMN_TB_VTB_Authorization_001(in verdicttype v_vct)
    runs on SECC_Tester
    return verdicttype {

    var EVSEProcessingType v_eVSEprocessing := ongoing_WaitingForCustomerInteraction;
    var charstring v_message;

    if((PICS_SECC_CMN_EIMDone == v2gAuthorization or not vc_testCaseSpecific)) {
        var boolean v_result := f_SECC_CMN_EIMIdentification();
    }

    while (v_eVSEprocessing == ongoing_WaitingForCustomerInteraction){
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(m_CMN_CMN_AuthorizationReq_001)));

        alt {
            [pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_001(vc_SessionID),
                md_CMN_BodyType_001(
                    md_CMN_CMN_AuthorizationRes_001(
                        oK, finished)))) {

                v_eVSEprocessing := finished;
                tc_V2G_EVCC_Msg_Timer.stop;
                tc_V2G_EVCC_Ongoing_Timer.stop;
                setverdict(pass,"AuthorizationRes message was correct.");

                if((PICS_SECC_CMN_EIMDone == v2gAuthorization or not vc_testCaseSpecific)) {
                    f_SECC_setEimStatus(true);
                }
            }

            and (PICS_CMN_CMN_IEC61851 and PICS_SECC_CMN_ActivateEIMHandling)
            [pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_001(vc_SessionID),
                md_CMN_BodyType_001(
                    md_CMN_CMN_AuthorizationRes_001(
                        oK, ongoing_WaitingForCustomerInteraction)))) {

                if(tc_V2G_EVCC_Ongoing_Timer.running==false){
                    tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
                }
            }

            [ a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
                "terminated by the SUT."){
                v_eVSEprocessing := finished;
                break;
            }

            [ a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                {authorizationRes := ?}) {
                v_eVSEprocessing := finished;
                break;
            }
        }
    }
}

```



```

[isError()] a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.") {
    v_eVSEprocessing := finished;
    break;
}
[] a_SECC_Unexpected_Message_Content(v_vct, "Received AuthorizationRes message, " &
    "but with invalid data content.") {
    v_eVSEprocessing := finished;
    break;
}
[] a_SECC_Timeout(v_vct){
    v_eVSEprocessing := finished;
    break;
}
[]tc_V2G_EVCC_Ongoing_Timer.timeout{
    setverdict(v_vct,"V2G_EVCC_Ongoing_Timer has expired. AuthorizationRes failed, "
&
    "EVSEProcessing := Finished was not sent in time.");
    v_eVSEprocessing := finished;
    break;
}
}
}
return getverdict;
}

// PnC
function f_SECC_CMN_TB_VTB_Authorization_002(in verdicttype v_vct)
    runs on SECC_Tester
    return verdicttype {

var EVSEProcessingType v_eVSEprocessing := ongoing;
var charstring v_message;
var integer v_cnt := 1;
var charstring v_privateKey;

while (v_eVSEprocessing == ongoing){
    if(v_cnt == 1) {
        var BodyType myAuthorizationBodyType := md_CMN_BodyType_001(
            md_CMN_CMN_AuthorizationReq_002(
                "authorizationReq", vc_GenChallenge));

        if(isbound(vc_contractPrivateKey)) {
            v_privateKey := vc_contractPrivateKey;
        } else {
            v_privateKey := fx_loadPrivateKey("CRT_CONTRACT_LEAF_VALID");
        }

        var Signature aSignature := md_CMN_Signature_001(
            md_CMN_SignedInfoType_001("authorizationReq",
                f_generateDigestFromBodyType(
                    myAuthorizationBodyType)), v_privateKey);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_003(
                vc_SessionID, aSignature),
            myAuthorizationBodyType));
    }
    else {
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                m_CMN_CMN_AuthorizationReq_001)));
    }
    v_cnt := v_cnt + 1;
    alt {
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_CMN_CMN_AuthorizationRes_001(
                    oK, finished)))) {
            v_eVSEprocessing := finished;
            tc_V2G_EVCC_Msg_Timer.stop;
            tc_V2G_EVCC_Ongoing_Timer.stop;
            setverdict(pass,"AuthorizationRes message was correct.");
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(

```

```

md_CMN_MessageHeaderType_001(vc_SessionID),
md_CMN_BodyType_001(
md_CMN_CMN_AuthorizationRes_001(
oK, ongoing)))) {

    if(tc_V2G_EVCC_Ongoing_Timer.running==false){
        tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
    }
}
[] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly" &
    " terminated by the SUT."){
    v_eVSEprocessing := finished;
    break;
}
[] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
    {authorizationRes := ?}) {
    v_eVSEprocessing := finished;
    break;
}
[isError()] a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.") {
    v_eVSEprocessing := finished;
    break;
}
[] a_SECC_Unexpected_Message_Content(v_vct, "Received AuthorizationRes message, " &
    "but with invalid data content.") {
    v_eVSEprocessing := finished;
    break;
}
[] a_SECC_Timeout(v_vct){
    v_eVSEprocessing := finished;
    break;
}
[]tc_V2G_EVCC_Ongoing_Timer.timeout{
    setverdict(v_vct,"V2G_EVCC_Ongoing_Timer has expired. AuthorizationRes failed, "
&
    "EVSEProcessing := Finished was not sent in time.");
    v_eVSEprocessing := finished;
    break;
}
}
}
return getverdict;
}

// EIM & PnC
function f_SECC_CMN_TB_VTB_Authorization_003() runs on SECC_Tester return verdictttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                mdw_SECC_CMN_ServiceDiscoveryRes_001(
                    fAILED_SequenceError)))) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                "'fAILED_SequenceError' is successful.");

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                e_getPortStatus)) to "local";

```

```

tc_CMN_TCP_Connection_Termination_Timer.start(
par_CMN_TCP_Connection_Termination_Timeout);
alt{
  [isShutdownOSC == false]
  a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
    isShutdownOSC := true;
    repeat;
  }
  [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
    "terminated by the SUT.") {
    if(isShutdownOSC == false) {
      logWarn("SECC oscillator was not shut down before the " &
        "TCP connection was terminated.");
      pt_HAL_61851_Internal_Port.clear;
    }
    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
      "but TCP connection was " &
      ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
  }
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
  "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
  {serviceDiscoveryRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
  "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
  "but with invalid data content.");
[] a_SECC_Timeout(fail);
}
return getverdict;
}

// EIM
function f_SECC_CMN_TB_VTB_Authorization_004() runs on SECC_Tester return verdicttype {

  var charstring v_message;
  var boolean isShutdownOSC := false;

  f_SECC_setIsConfirmationFlagDC();
  f_SECC_changeValidFrequencyRange(0,0);
  f_SECC_changeValidDutyCycleRange(100,100);
  deactivate(vc_Default_IEC_61851_ListenerBehavior);

  tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
  pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(
      f_rnd_SessionID(1,429496)),
    md_CMN_BodyType_001(
      m_CMN_CMN_AuthorizationReq_001)));

  alt {
    []
    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
      isShutdownOSC := true;
      repeat;
    }
    []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
      md_CMN_MessageHeaderType_001(?),
      md_CMN_BodyType_001(
        md_CMN_CMN_AuthorizationRes_001(
          fAILED_UnknownSession, ?)))) {

      tc_V2G_EVCC_Msg_Timer.stop;
      setverdict(pass,"AuthorizationRes with responseCode " &
        "'fAILED_UnknownSession' is successful.");

      pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_getPortStatus)) to "local";
      tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);
      alt{
        [isShutdownOSC == false]
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
          isShutdownOSC := true;
          repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
          "terminated by the SUT.") {

```

```

        if(isShutdownOSC == false) {
            logWarn("SECC oscillator was not shut down before the " &
                "TCP connection was terminated.");}
            pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
            "but TCP connection was " &
            ">>NOT<< terminated by the SUT");
        [] a_SECC_TCP_Connection_Termination_Timeout();
    }
    [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
        "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
        {authorizationRes := ?});
    [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
        "but V2G_Message was expected.");
    [] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
        "but with invalid data content.");
    [] a_SECC_Timeout(fail);
}
return getverdict;
}

// PnC
function f_SECC_CMN_TB_VTB_Authorization_005() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var charstring v_privateKey;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);

    var BodyType myAuthorizationBodyType := md_CMN_BodyType_001(
        md_CMN_CMN_AuthorizationReq_002(
            "authorizationReq", vc_GenChallenge));

    if(isbound(vc_contractPrivateKey)) {
        v_privateKey := vc_contractPrivateKey;
    } else {
        v_privateKey := fx_loadPrivateKey("CRT_CONTRACT_LEAF_VALID");
    }

    var Signature aSignature := md_CMN_Signature_001(
        md_CMN_SignedInfoType_001("authorizationReq",
            f_generateDigestFromBodyType(myAuthorizationBodyType),
            v_privateKey);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_003(
            f_rnd_SessionID(1,429496), aSignature),
            myAuthorizationBodyType));

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(?),
            md_CMN_BodyType_001(
                md_CMN_CMN_AuthorizationRes_001(
                    fAILED_UnknownSession, ?)))) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass, "AuthorizationRes with responseCode " &
                "'fAILED_UnknownSession' is successful.");

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                e_getPortStatus)) to "local";
            tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
            alt{

```

```

[isShutdownOSC == false]
a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
    isShutdownOSC := true;
    repeat;
}
[] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
    "terminated by the SUT.") {
    if(isShutdownOSC == false) {
        logWarn("SECC oscillator was not shut down before the " &
            "TCP connection was terminated.");
        pt_HAL_61851_Internal_Port.clear;
    }
    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
        "but TCP connection was " &
        ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {authorizationRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
    "but with invalid data content.");
[] a_SECC_Timeout(fail);
}
return getverdict;
}

// PnC
function f_SECC_CMN_TB_VTB_Authorization_006() runs on SECC_Tester return verdicttype {
    var charstring v_message;
    var boolean isShutdownOSC := false;
    var charstring v_privateKey;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    vc_GenChallenge := fx_generateGenChallenge();

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);

    var BodyType myAuthorizationBodyType := md_CMN_BodyType_001(
        md_CMN_CMN_AuthorizationReq_002(
            "authorizationReq", vc_GenChallenge));

    if(isbound(vc_contractPrivateKey)) {
        v_privateKey := vc_contractPrivateKey;
    } else {
        v_privateKey := fx_loadPrivateKey("CRT_CONTRACT_LEAF_VALID");
    }

    var Signature aSignature := md_CMN_Signature_001(
        md_CMN_SignedInfoType_001("authorizationReq",
            f_generatedDigestFromBodyType(
                myAuthorizationBodyType)), v_privateKey);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_003(
            vc_SessionID, aSignature),
        myAuthorizationBodyType));

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(
                vc_SessionID),
            md_CMN_BodyType_001(
                md_CMN_CMN_AuthorizationRes_001(
                    FAILED_ChallengeInvalid, ?)))) {

```

```

tc_V2G_EVCC_Msg_Timer.stop;
setverdict(pass, "AuthorizationRes with responseCode " &
    "'FAILED_ChallengeInvalid' is successful.");

pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
    e_getPortStatus)) to "local";
tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);
alt{
    [isShutdownOSC == false]
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
    [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
        "terminated by the SUT.") {
        if(isShutdownOSC == false) {
            logWarn("SECC oscillator was not shut down before the " &
                "TCP connection was terminated.");}
        pt_HAL_61851_Internal_Port.clear;
    }
    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
        "but TCP connection was " &
        ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {authorizationRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
    "but with invalid data content.");
[] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_Authorization_007() runs on SECC_Tester return verdicttype {
    timer t_wait;
    var boolean isShutdownOSC := false;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout - par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_getPortStatus)) to "local";

    alt {
        [] t_wait.timeout;
    }

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        []
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
            "terminated by the SUT.") {
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                    "TCP connection was terminated.");}
            pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
            "but TCP connection was " &
            ">>NOT<< terminated by the SUT");
        [] a_SECC_TCP_Connection_Termination_Timeout();
    }
}

```

```

    }
    return getverdict;
}

// PnC
function f_SECC_CMN_TB_VTB_Authorization_008() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);

    var BodyType myAuthorizationBodyType := md_CMN_BodyType_001(
        md_CMN_CMN_AuthorizationReq_002(
            "authorizationReq", vc_GenChallenge));

    // load wrong private key
    var charstring v_invalidPrivateKey := fx_loadPrivateKey("CRT_OEM_LEAF_VALID");

    var Signature aSignature := md_CMN_Signature_001(
        md_CMN_SignedInfoType_001("authorizationReq",
            f_generateDigestFromBodyType(myAuthorizationBodyType)),
        v_invalidPrivateKey);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_003(
            vc_SessionID, aSignature),
        myAuthorizationBodyType));

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(
                vc_SessionID),
            md_CMN_BodyType_001(
                md_CMN_CMN_AuthorizationRes_001(
                    fFAILED_SignatureError, ?)))) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass, "AuthorizationRes with responseCode " &
                "'fFAILED_SignatureError' is successful.");

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                e_getPortStatus)) to "local";
            tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
            alt{
                [isShutdownOSC == false]
                a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                    isShutdownOSC := true;
                    repeat;
                }
                [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                    "terminated by the SUT.") {
                    if(isShutdownOSC == false) {
                        logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");
                        pt_HAL_61851_Internal_Port.clear;
                    }
                }
                [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                    "but TCP connection was " &
                    ">>NOT<< terminated by the SUT");
                [] a_SECC_TCP_Connection_Termination_Timeout();
            }
        }
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
            "terminated by the SUT.");
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
            {authorizationRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &

```



```

        "but V2G_Message was expected.");
    [] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
        "but with invalid data content.");
    [] a_SECC_Timeout(fail);
}
return getverdict;
}

// PnC
function f_SECC_CMN_TB_VTB_Authorization_009() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var charstring v_privateKey;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);

    var BodyType myAuthorizationBodyType := md_CMN_BodyType_001(
        md_CMN_CMN_AuthorizationReq_002(
            "authorizationReq", vc_GenChallenge));

    if(isbound(vc_contractPrivateKey)) {
        v_privateKey := vc_contractPrivateKey;
    } else {
        v_privateKey := fx_loadPrivateKey("CRT_CONTRACT_LEAF_VALID");
    }

    var Signature aSignature := md_CMN_Signature_001(
        md_CMN_SignedInfoType_001("authorizationReq",
            f_generateDigestFromBodyType(myAuthorizationBodyType)),
        v_privateKey);

    // change genChallenge
    vc_GenChallenge:=fx_generateGenChallenge();
    myAuthorizationBodyType := md_CMN_BodyType_001(
        md_CMN_CMN_AuthorizationReq_002(
            "authorizationReq", vc_GenChallenge));

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_003(
            vc_SessionID, aSignature),
        myAuthorizationBodyType));

    alt {
    []
    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
    }
    []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(
            vc_SessionID),
        md_CMN_BodyType_001(
            md_CMN_CMN_AuthorizationRes_001(
                fAILED_SignatureError, ?)))) {

        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(pass,"AuthorizationRes with responseCode " &
            "'fAILED_SignatureError' is successful.");

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
            e_getPortStatus)) to "local";
        tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
        alt{
            [isShutdownOSC == false]
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                "terminated by the SUT.") {
                if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &

```

```

        "TCP connection was terminated.");}
        pt_HAL_61851_Internal_Port.clear;
    }
    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
        "but TCP connection was " &
        ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {authorizationRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
    "but with invalid data content.");
[] a_SECC_Timeout(fail);
}
return getverdict;
}

// EIM
function f_SECC_CMN_TB_VTB_Authorization_010() runs on SECC_Tester return verdicttype {

    var EVSEProcessingType v_eVSEprocessing := ongoing_WaitingForCustomerInteraction;
    var charstring v_message;
    var boolean isShutdownOSC := false;

    while (v_eVSEprocessing == ongoing_WaitingForCustomerInteraction){
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(m_CMN_CMN_AuthorizationReq_001)));
        alt {
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_001(vc_SessionID),
                md_CMN_BodyType_001(
                    md_CMN_CMN_AuthorizationRes_001(
                        oK, finished)))) {
                v_eVSEprocessing := finished;
                tc_V2G_EVCC_Msg_Timer.stop;
                tc_V2G_EVCC_Ongoing_Timer.stop;
                setverdict(inconc) "No EIM authorization was initiated before.";
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_001(vc_SessionID),
                md_CMN_BodyType_001(
                    md_CMN_CMN_AuthorizationRes_001(
                        oK, ongoing_WaitingForCustomerInteraction)))) {
                if(tc_V2G_SECC_Ongoing_Timer.running==false){
                    tc_V2G_SECC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Performance_Timeout -
                        par_CMN_Transmission_Delay);
                }
            }
            [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
                "terminated by the SUT."){
                v_eVSEprocessing := finished;
                break;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                {authorizationRes := ?}) {
                v_eVSEprocessing := finished;
                break;
            }
            [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
                "but V2G_Message was expected.") {
                v_eVSEprocessing := finished;
                break;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
                "but with invalid data content.") {
                v_eVSEprocessing := finished;
                break;
            }
        }
        [] a_SECC_Timeout(fail){
            v_eVSEprocessing := finished;
            break;
        }
    }
}

```

```

    }
    []tc_V2G_SECC_Ongoing_Timer.timeout{

        f_SECC_setIsConfirmationFlagDC();
        f_SECC_changeValidFrequencyRange(0,0);
        f_SECC_changeValidDutyCycleRange(100,100);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);
        tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            []
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                "terminated by the SUT.") {
                if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
                    pt_HAL_61851_Internal_Port.clear;
                }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                "but TCP connection was " &
                ">>NOT<< terminated by " &
                "the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout();
        }
        v_eVSEprocessing := finished;
        break;
    }
}
}
return getverdict;
}

// PnC
function f_SECC_CMN_TB_VTB_Authorization_011() runs on SECC_Tester return verdicttype {

    var EVSEProcessingType v_eVSEprocessing := ongoing;
    var charstring v_message;
    var integer v_cnt := 1;
    var charstring v_privateKey;
    var boolean isShutdownOSC := false;

    while (v_eVSEprocessing == ongoing){
        if(v_cnt == 1) {
            var BodyType myAuthorizationBodyType := md_CMN_BodyType_001(
                md_CMN_CMN_AuthorizationReq_002(
                    "authorizationReq", vc_GenChallenge));

            if(isbound(vc_contractPrivateKey)) {
                v_privateKey := vc_contractPrivateKey;
            } else {
                v_privateKey := fx_loadPrivateKey("CRT_CONTRACT_LEAF_VALID");
            }

            var Signature aSignature := md_CMN_Signature_001(
                md_CMN_SignedInfoType_001("authorizationReq",
                    f_generateDigestFromBodyType(
                        myAuthorizationBodyType)), v_privateKey);

            tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_003(
                    vc_SessionID, aSignature),
                myAuthorizationBodyType));
        }
        else {
            tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_001(vc_SessionID),
                md_CMN_BodyType_001(
                    m_CMN_CMN_AuthorizationReq_001)));
        }
        v_cnt := v_cnt + 1;
        alt {
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(

```

```

md_CMN_MessageHeaderType_001(vc_SessionID),
md_CMN_BodyType_001(
md_CMN_CMN_AuthorizationRes_001(
oK, finished)))) {

v_eVSEprocessing := finished;
tc_V2G_EVCC_Msg_Timer.stop;
tc_V2G_EVCC_Ongoing_Timer.stop;
setverdict(inconc, "No authorization should be triggered by the SUT before.");
}
[]pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
md_CMN_MessageHeaderType_001(vc_SessionID),
md_CMN_BodyType_001(
md_CMN_CMN_AuthorizationRes_001(
oK, ongoing)))) {

if(tc_V2G_SECC_Ongoing_Timer.running==false){
tc_V2G_SECC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Performance_Timeout -
par_CMN_Transmission_Delay);
}
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly" &
" terminated by the SUT."){
v_eVSEprocessing := finished;
break;
}
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
{authorizationRes := ?}) {
v_eVSEprocessing := finished;
break;
}
[]isError() a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
"but V2G_Message was expected.") {
v_eVSEprocessing := finished;
break;
}
[] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
"but with invalid data content.") {
v_eVSEprocessing := finished;
break;
}
[] a_SECC_Timeout(fail){
v_eVSEprocessing := finished;
break;
}
[]tc_V2G_SECC_Ongoing_Timer.timeout{

f_SECC_setIsConfirmationFlagDC();
f_SECC_changeValidFrequencyRange(0,0);
f_SECC_changeValidDutyCycleRange(100,100);
deactivate(vc_Default_IEC_61851_ListenerBehavior);
tc_CMN_TCP_Connection_Termination_Timer.start(
par_CMN_TCP_Connection_Termination_Timeout);

alt {
[]
a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
isShutdownOSC := true;
repeat;
}
[] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
"terminated by the SUT.") {
if(isShutdownOSC == false) {
logWarn("SECC oscillator was not shut down before the " &
"TCP connection was terminated.");}
pt_HAL_61851_Internal_Port.clear;
}
[] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
"but TCP connection was " &
">>NOT<< terminated by " &
"the SUT");
[] a_SECC_TCP_Connection_Termination_Timeout();
}
v_eVSEprocessing := finished;
break;
}
}
return getverdict;
}

```

}

I.1.2.7 SECC functions for ChargeParameterDiscovery

```

module TestBehavior_SECC_ChargeParameterDiscovery {

    import from Timer_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from ComponentsAndPorts all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from Templates_SECC_ChargeParameterDiscovery all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_SECC_ServiceDiscovery all;
    import from Templates_CMN_V2GMessage all;
    import from Templates_CMN_ChargingStatus all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Pics_15118_2 all;
    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_ServiceDiscovery all;
    import from Templates_CMN_PowerDelivery all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from XSDAUX all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_Security all;
    import from http_www_w3_org_2000_09_xmlsig all;
    import from Services_Security all;
    import from Pixit_15118_2 all;
    import from Timer_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118 all;

    function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(in verdicttype v_vct)
        runs on SECC_Tester
        return verdicttype {

            var EVSEProcessingType v_EVSEprocessing := ongoing;
            var V2G_Message v_responseMessage;
            var charstring v_message;
            var float v_pMax;
            var boolean isSignatureOk := true;

            if(f_checkRequestedEnergyTransferMode(vc_requestedEnergyTransferModeAC,
                                                vc_supportedEnergyTransferMode)) {

                while (v_EVSEprocessing == ongoing){

                    tc_V2G_EVCC_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

                    if(PICS_SECC_CMN_DepartureTime) {

                        if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                    md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                        omit, vc_requestedEnergyTransferModeAC,
                                        vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                        vc_EVMaxCurrent, vc_EVMinCurrent))));
                        } else {
                            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                    md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                        par_CMN_MaxEntriesSAScheduleTuple,
                                        vc_requestedEnergyTransferModeAC,
                                        vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                        vc_EVMaxCurrent, vc_EVMinCurrent))));
                        }
                    }
                }
            } else {

                if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),

```

```

md_CMN_BodyType_001(
md_SECC_AC_ChargeParameterDiscoveryReq_001(
omit, vc_requestedEnergyTransferModeAC,
omit, vc_EAmount, vc_EVMaxVoltage,
vc_EVMaxCurrent, vc_EVMinCurrent)))));
} else {
pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
md_CMN_MessageHeaderType_001(vc_SessionID),
md_CMN_BodyType_001(
md_SECC_AC_ChargeParameterDiscoveryReq_001(
par_CMN_MaxEntriesSAScheduleTuple,
vc_requestedEnergyTransferModeAC,
omit, vc_EAmount, vc_EVMaxVoltage,
vc_EVMaxCurrent, vc_EVMinCurrent)))));
}
}
alt {
[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
md_CMN_MessageHeaderType_002(vc_SessionID),
md_CMN_BodyType_001(
mdw_SECC_AC_ChargeParameterDiscoveryRes_001(
finished, md_CMN_AC_EVSEStatus_001(?,?,?),
*))))
-> value v_responseMessage {

v_EVSEprocessing := finished;
vc_SASchedule_List := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.sASchedules.
sAScheduleList;
vc_PMaxSchedule := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.sASchedules.sAScheduleList.
sAScheduleTuple_list[0].pMaxSchedule;
vc_SASchedules := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.sASchedules;
vc_SAScheduleTupleId := vc_SASchedule_List.sAScheduleTuple_list[0].
sAScheduleTupleID;
vc_EVSENominalVoltage := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.evSEChargeParameter.
aC_EVSEChargeParameter.evSENominalVoltage;
f_check_PhysicalValue_valueRange(vc_EVSENominalVoltage, v,
cc_maximum_value_physicalValue_case2,
v_vct);
vc_EVSEMaxCurrent := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.evSEChargeParameter.
aC_EVSEChargeParameter.evSEMaxCurrent;
f_check_PhysicalValue_valueRange(vc_EVSEMaxCurrent, a,
cc_maximum_value_physicalValue_case1,
v_vct);

tc_V2G_EVCC_Msg_Timer.stop;
tc_V2G_EVCC_Ongoing_Timer.stop;
if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
evSEChargeParameter.aC_EVSEChargeParameter.aC_EVSEStatus.
evSENotification != none) {
logWarn("The field 'evSENotification' should be set to 'none'.");
}
if(vc_requestedEnergyTransferModeAC == aC_three_phase_core) {
v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent) * 3.0;
}
else {
v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent);
}

f_checkACPMMaxScheduleEntries(vc_SASchedule_List, v_pMax, v_vct);
f_checkPMaxSchedulesStructure(vc_SASchedule_List,
cc_maximum_value_physicalValue_case4,
PICS_SECC_CMN_MaxEntriesSAScheduleTuple,
v_vct);

if(getverdict == pass) {
f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime,
vc_SASchedules, v_vct);
}

if(getverdict == pass) {
setverdict(pass, "ChargeParameterDiscoveryRes message " &
"was correct.");
}
}
[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(

```

```

md_CMN_MessageHeaderType_001(vc_SessionID),
md_CMN_BodyType_001(
mdw_SECC_AC_ChargeParameterDiscoveryRes_002(
md_CMN_AC_EVSEStatus_001(?,?,?))))
-> value v_responseMessage {

if(tc_V2G_EVCC_Ongoing_Timer.running==false){
    tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
}
if(v_responseMessage.body.bodyElement.
    chargeParameterDiscoveryRes.evSEChargeParameter.ac_EVSEChargeParameter.
    ac_EVSEStatus.evSENotification != none_) {
    logWarn("The field 'evSENotification' should be set to 'none_'.");
}
}
[] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
    "terminated by the SUT.") {
    v_EVSEprocessing := finished;
    break;
}
[] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
    {chargeParameterDiscoveryRes := ?}) {
    v_EVSEprocessing := finished;
    break;
}
[isError()] a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.") {
    v_EVSEprocessing := finished;
    break;
}
[] a_SECC_Unexpected_Message_Content(v_vct, "Received ChargeParameterDiscoveryRes "
    "message, but with invalid " &
    "data content.") {
    v_EVSEprocessing := finished;
    break;
}
[] a_SECC_Timeout(v_vct) {
    v_EVSEprocessing := finished;
    break;
}
[] tc_V2G_EVCC_Ongoing_Timer.timeout{
    v_EVSEprocessing := finished;
    setverdict(v_vct, "EVSEProcessing = Finished was not sent in time.");
    break;
}
}
}
}
else {setverdict(inconc, "Requested energy transfer mode is not supported " &
    "by the SECC.");}
return getverdict;
}

function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_002() runs on SECC_Tester
    return verdicttype {

var charstring v_message;
var boolean isShutdownOSC := false;

f_SECC_setIsConfirmationFlagDC();
f_SECC_changeValidFrequencyRange(0,0);
f_SECC_changeValidDutyCycleRange(100,100);

if(f_checkRequestedEnergyTransferMode(vc_requestedEnergyTransferModeAC,
    vc_supportedEnergyTransferMode)) {

tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);
pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(
        f_rnd_SessionID(1,429496)),
    md_CMN_BodyType_001(
        md_SECC_AC_ChargeParameterDiscoveryReq_001(
            omit, vc_requestedEnergyTransferModeAC, vc_DepartureTime,
            vc_EAmount, vc_EVMaxVoltage,
            vc_EVMaxCurrent, vc_EVMinCurrent))));

alt {
[]

```



```

a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
    isShutdownOSC := true;
    repeat;
}
[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_002(?),
    md_CMN_BodyType_001(
        mdw_SECC_CMN_ChargeParameterDiscoveryRes_001(
            FAILED_UnknownSession)))) {

    tc_V2G_EVCC_Msg_Timer.stop;
    setverdict(pass,"ChargeParameterDiscoveryRes with responseCode " &
        "'FAILED_UnknownSession' is successful.");

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_getPortStatus)) to "local";
    tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);
    alt{
        [isShutdownOSC == false]
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP Connection was " &
            "terminated by the SUT.") {
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                    "TCP connection was terminated.");
                pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                "but TCP connection was " &
                ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout();
        }
    }
    [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
        "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
        {chargeParameterDiscoveryRes := ?});
    [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
        "but V2G_Message was expected.");
    [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
        "message, but with invalid " &
        "data content.");
    [] a_SECC_Timeout(fail);
}
}
else {setverdict(inconc,"Requested energy transfer mode is not supported by the SECC.");}
return getverdict;
}

function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_003() runs on SECC_Tester
    return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);

    vc_requestedEnergyTransferModeAC := dC_extended;
    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_SECC_AC_ChargeParameterDiscoveryReq_001(
                omit, vc_requestedEnergyTransferModeAC,
                vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                vc_EVMaxCurrent, vc_EVMinCurrent))));

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
    }
}

```

```

[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_002(vc_SessionID),
    md_CMN_BodyType_001(
        mdw_SECC_CMN_ChargeParameterDiscoveryRes_001(
            fAILED_WrongEnergyTransferMode)))) {

    tc_V2G_EVCC_Msg_Timer.stop;
    setverdict(pass,"ChargeParameterDiscoveryRes with responseCode " &
        "'fAILED_WrongEnergyTransferMode' is successful.");

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_getPortStatus)) to "local";
    tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);
    alt{
        [isShutdownOSC == false]
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
            "terminated by the SUT.");
        if(isShutdownOSC == false) {
            logWarn("SECC oscillator was not shut down before the " &
                "TCP connection was terminated.");
            pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
            "but TCP connection was " &
            ">>NOT<< terminated by the SUT");
        [] a_SECC_TCP_Connection_Termination_Timeout();
    }
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {chargeParameterDiscoveryRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
    "message, but with invalid " &
    "data content.");

[] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_004() runs on SECC_Tester
    return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_SECC_AC_ChargeParameterDiscoveryReq_002(
                vc_requestedEnergyTransferModeAC))));

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_002(vc_SessionID),
            md_CMN_BodyType_001(
                mdw_SECC_CMN_ChargeParameterDiscoveryRes_001(
                    fAILED_WrongChargeParameter)))) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"ChargeParameterDiscoveryRes with responseCode " &
                "'fAILED_WrongChargeParameter' is successful.");

```

```

pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
    e_getPortStatus)) to "local";
tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);
alt{
    [isShutdownOSC == false]
    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
    }
    [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
        "terminated by the SUT.") {
        if(isShutdownOSC == false) {
            logWarn("SECC oscillator was not shut down before the " &
                "TCP connection was terminated.");}
            pt_HAL_61851_Internal_Port.clear;
        }
    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
        "but TCP connection was " &
        ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {chargeParameterDiscoveryRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
    "message, but with invalid " &
    "data content.");

[] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005() runs on SECC_Tester
    return verdicttype {

    var EVSEProcessingType v_EVSEprocessing := ongoing;
    var V2G_Message v_responseMessage;
    var charstring v_message;
    var float v_pMax;
    var boolean isSignatureOk := true;

    if(f_checkRequestedEnergyTransferMode(vc_requestedEnergyTransferModeAC,
        vc_supportedEnergyTransferMode)) {

        while (v_EVSEprocessing == ongoing){

            tc_V2G_EVCC_Msg_Timer.start(
                par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

            if(PICS_SECC_CMN_DepartureTime) {

                if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                omit, vc_requestedEnergyTransferModeAC,
                                vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                vc_EVMaxCurrent, vc_EVMinCurrent))));
                } else {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                par_CMN_MaxEntriesSAScheduleTuple,
                                vc_requestedEnergyTransferModeAC,
                                vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                vc_EVMaxCurrent, vc_EVMinCurrent))));
                }
            } else {

                if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(

```

```

md_CMN_MessageHeaderType_001(vc_SessionID),
md_CMN_BodyType_001(
md_SECC_AC_ChargeParameterDiscoveryReq_001(
omit, vc_requestedEnergyTransferModeAC,
omit, vc_EAmount, vc_EVMaxVoltage,
vc_EVMaxCurrent, vc_EVMinCurrent))));
} else {
pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
md_CMN_MessageHeaderType_001(vc_SessionID),
md_CMN_BodyType_001(
md_SECC_AC_ChargeParameterDiscoveryReq_001(
par_CMN_MaxEntriesSAScheduleTuple,
vc_requestedEnergyTransferModeAC,
omit, vc_EAmount, vc_EVMaxVoltage,
vc_EVMaxCurrent, vc_EVMinCurrent))));
}
}
alt {
[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
md_CMN_MessageHeaderType_001(vc_SessionID),
md_CMN_BodyType_001(
mdw_SECC_AC_ChargeParameterDiscoveryRes_001(
finished, md_CMN_AC_EVSEStatus_001(?, ?, ?),
*))))
-> value v_responseMessage {

v_EVSEprocessing := finished;
vc_SASchedule_List := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.sASchedules.
sAScheduleList;
vc_PMaxSchedule := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.sASchedules.
sAScheduleList.sAScheduleTuple_list[0].
pMaxSchedule;
vc_SASchedules := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.sASchedules;
vc_SAScheduleTupleId := vc_SASchedule_List.sAScheduleTuple_list[0].
sAScheduleTupleID;
vc_EVSENominalVoltage := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.
evSEChargeParameter.aC_EVSEChargeParameter.
evSENominalVoltage;
f_check_PhysicalValue_valueRange(vc_EVSENominalVoltage, v,
cc_maximum_value_physicalValue_case2,
fail);
vc_EVSEMaxCurrent := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.evSEChargeParameter.
aC_EVSEChargeParameter.evSEMaxCurrent;
f_check_PhysicalValue_valueRange(vc_EVSEMaxCurrent, a,
cc_maximum_value_physicalValue_case1,
fail);

tc_V2G_EVCC_Msg_Timer.stop;
tc_V2G_EVCC_Ongoing_Timer.stop;
if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
evSEChargeParameter.aC_EVSEChargeParameter.aC_EVSEStatus.
evSENotification != none) {
logWarn("The field 'evSENotification' should be set to 'none'.");
}
if(vc_requestedEnergyTransferModeAC == aC_three_phase_core) {
v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent) * 3.0;
}
else {
v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent);
}
f_checkACPMMaxScheduleEntries(vc_SASchedule_List, v_pMax, fail);
f_checkPMaxSchedulesStructure(vc_SASchedule_List,
cc_maximum_value_physicalValue_case4,
PICS_SECC_CMN_MaxEntriesSAScheduleTuple,
fail);

if(getverdict == pass) {
f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime, vc_SASchedules,
fail);
}
if(getverdict == pass) {
if(f_compareToPreviousSAScheduleTuple(vc_SASchedule_List,
vc_SAScheduleTupleId)) {

```

```

        setverdict(pass, "ChargeParameterDiscoveryRes message was correct. " &
            "A valid SAScheduleTuple was received after " &
            "paused session.");
    }
    else {
        setverdict(fail, "Invalid SAScheduleTuple was received after " &
            "paused session.");
    }
}

[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(vc_SessionID),
    md_CMN_BodyType_001(
        mdw_SECC_AC_ChargeParameterDiscoveryRes_002(
            md_CMN_AC_EVSEStatus_001(?,?,?))))
-> value v_responseMessage {

    if(tc_V2G_EVCC_Ongoing_Timer.running==false){
        tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
    }
    if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
        evSEChargeParameter.ac_EVSEChargeParameter.ac_EVSEStatus.
        evSENotification != none) {
        logWarn("The field 'evSENotification' should be set to 'none_'.");
    }
}

[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.") {
    v_EVSEprocessing := finished;
    break;
}

[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {chargeParameterDiscoveryRes := ?}) {
    v_EVSEprocessing := finished;
    break;
}

[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.") {
    v_EVSEprocessing := finished;
    break;
}

[] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
    "message, but with invalid " &
    "data content.") {
    v_EVSEprocessing := finished;
    break;
}

[] a_SECC_Timeout(fail) {
    v_EVSEprocessing := finished;
    break;
}

[] tc_V2G_EVCC_Ongoing_Timer.timeout{
    v_EVSEprocessing := finished;
    setverdict(fail, "EVSEProcessing = Finished was not sent in time.");
    break;
}

}
else {setverdict(inconc, "Requested energy transfer mode is not supported by the SECC.");}
return getverdict;
}

function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(in verdicttype v_vct)
    runs on SECC_Tester
    return verdicttype {

    var EVSEProcessingType v_EVSEprocessing := ongoing;
    var V2G_Message v_responseMessage;
    var charstring v_message;
    var float v_pMax;
    var boolean isSignatureOk := true;

    if(f_checkRequestedEnergyTransferMode(vc_requestedEnergyTransferModeAC,
        vc_supportedEnergyTransferMode)) {

        while (v_EVSEprocessing == ongoing){

            tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

```

```

if(PICS_SECC_CMN_DepartureTime) {

    if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_SECC_AC_ChargeParameterDiscoveryReq_001(
                    omit, vc_requestedEnergyTransferModeAC,
                    vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                    vc_EVMaxCurrent, vc_EVMinCurrent))));
    } else {

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_SECC_AC_ChargeParameterDiscoveryReq_001(
                    par_CMN_MaxEntriesSAScheduleTuple,
                    vc_requestedEnergyTransferModeAC,
                    vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                    vc_EVMaxCurrent, vc_EVMinCurrent))));
    }
} else {

    if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_SECC_AC_ChargeParameterDiscoveryReq_001(
                    omit, vc_requestedEnergyTransferModeAC,
                    omit, vc_EAmount, vc_EVMaxVoltage,
                    vc_EVMaxCurrent, vc_EVMinCurrent))));
    } else {

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_SECC_AC_ChargeParameterDiscoveryReq_001(
                    par_CMN_MaxEntriesSAScheduleTuple,
                    vc_requestedEnergyTransferModeAC,
                    omit, vc_EAmount, vc_EVMaxVoltage,
                    vc_EVMaxCurrent, vc_EVMinCurrent))));
    }
}

alt {
    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_002(vc_SessionID),
        md_CMN_BodyType_001(
            mdw_SECC_AC_ChargeParameterDiscoveryRes_001(
                finished, md_CMN_AC_EVSEStatus_001(?,?,?),?)))
        -> value v_responseMessage {

            v_EVSEprocessing := finished;
            vc_SASchedule_List := v_responseMessage.body.bodyElement.
                chargeParameterDiscoveryRes.sASchedules.
                sAScheduleList;
            vc_PMaxSchedule := v_responseMessage.body.bodyElement.
                chargeParameterDiscoveryRes.sASchedules.sAScheduleList.
                sAScheduleTuple_list[0].pMaxSchedule;
            vc_SASchedules := v_responseMessage.body.bodyElement.
                chargeParameterDiscoveryRes.sASchedules;
            vc_SAScheduleTupleId := vc_SASchedule_List.sAScheduleTuple_list[0].
                sAScheduleTupleID;
            vc_EVSENominalVoltage := v_responseMessage.body.bodyElement.
                chargeParameterDiscoveryRes.evSEChargeParameter.
                aC_EVSEChargeParameter.evSENominalVoltage;
            f_check_PhysicalValue_valueRange(vc_EVSENominalVoltage, v,
                cc_maximum_value_physicalValue_case2,
                v_vct);
            vc_EVSEMaxCurrent := v_responseMessage.body.bodyElement.
                chargeParameterDiscoveryRes.evSEChargeParameter.
                aC_EVSEChargeParameter.evSEMaxCurrent;
            f_check_PhysicalValue_valueRange(vc_EVSEMaxCurrent, a,
                cc_maximum_value_physicalValue_case1,
                v_vct);

            tc_V2G_EVCC_Msg_Timer.stop;
            tc_V2G_EVCC_Ongoing_Timer.stop;
        }
}

```

```

if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
  eVSEChargeParameter.aC_EVSEChargeParameter.aC_EVSEStatus.
  eVSENotification != none_) {
  logWarn("The field 'eVSENotification' should be set to 'none_'.");
}
if(vc_requestedEnergyTransferModeAC == aC_three_phase_core) {
  v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
    f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent) * 3.0;
}
else {
  v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
    f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent);
}

f_checkACPMMaxScheduleEntries(vc_SASchedule_List,
  v_pMax, v_vct);

if(getverdict == pass) {
  f_checkPMaxSchedulesStructure(vc_SASchedule_List,
    cc_maximum_value_physicalValue_case4,
    PICS_SECC_CMN_MaxEntriesSAScheduleTuple, v_vct);
}
if(getverdict == pass) {
  f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime,
    vc_SASchedules, v_vct);
}
if(getverdict == pass) {
  f_checkSalesTariffStructure(vc_SASchedule_List,
    PICS_SECC_CMN_MaxEntriesSAScheduleTuple, v_vct);
}
if(getverdict == pass) {
  f_checkSalesTariffTimeIntervals(vc_DepartureTime,
    vc_SASchedules, v_vct);
}

if(PICS_CMN_CMN_IdentificationMode == pnC) {
  var idDigestValueMap resultIdDigestValueMap :=
    f_calculateIdDigestMapFromBodyType(v_responseMessage.body);
  if(lengthof(resultIdDigestValueMap) > 0) {
    var Signature authSignature := v_responseMessage.header.signature_;
    var BodyType chargeParameterResBodyType := v_responseMessage.body;

    isSignatureOk := f_verifySignature(
      chargeParameterResBodyType,
      authSignature,
      vc_contractSignatureCertChain.
      subCertificates.
      x509Certificate_list[0], v_vct);

    if(isSignatureOk) {
      setverdict(pass, "Valid signature of the sales " &
        "tariff was received.");
    }
    else {
      setverdict(v_vct, "Invalid signature of the sales " &
        "tariff was received.");
    }
  }
}
if(getverdict == pass) {
  setverdict(pass, "ChargeParameterDiscoveryRes message " &
    "was correct.");
}
}

[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
  md_CMN_MessageHeaderType_001(vc_SessionID),
  md_CMN_BodyType_001(
    mdw_SECC_AC_ChargeParameterDiscoveryRes_002(
      md_CMN_AC_EVSEStatus_001(?,?,?))))
  -> value v_responseMessage {

  if(tc_V2G_EVCC_Ongoing_Timer.running==false){
    tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
  }
  if(v_responseMessage.body.bodyElement.
    chargeParameterDiscoveryRes.eVSEChargeParameter.aC_EVSEChargeParameter.
    aC_EVSEStatus.eVSENotification != none_) {
    logWarn("The field 'eVSENotification' should be set to 'none_'.");
  }
}

[] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
  "terminated by the SUT.") {

```



```

        v_EVSEprocessing := finished;
        break;
    }
    [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                     {chargeParameterDiscoveryRes := ?}) {
        v_EVSEprocessing := finished;
        break;
    }
    [isError()] a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes received, " &
                                             "but V2G_Message was expected.") {
        v_EVSEprocessing := finished;
        break;
    }
    [] a_SECC_Unexpected_Message_Content(v_vct, "Received ChargeParameterDiscoveryRes " &
                                              "message, but with invalid " &
                                              "data content.") {
        v_EVSEprocessing := finished;
        break;
    }
    [] a_SECC_Timeout(v_vct) {
        v_EVSEprocessing := finished;
        break;
    }
    [] tc_V2G_EVCC_Ongoing_Timer.timeout{
        v_EVSEprocessing := finished;
        setverdict(v_vct, "EVSEProcessing = Finished was not sent in time.");
        break;
    }
}
}
}
else {setverdict(inconc, "Requested energy transfer mode is not supported by the SECC.");}
return getverdict;
}

function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007() runs on SECC Tester
    return verdicttype {

    var EVSEProcessingType v_EVSEprocessing := ongoing;
    var V2G_Message v_responseMessage;
    var charstring v_message;
    var float v_pMax;
    var boolean isSignatureOk := true;

    if(f_checkRequestedEnergyTransferMode(vc_requestedEnergyTransferModeAC,
                                          vc_supportedEnergyTransferMode)) {

        while (v_EVSEprocessing == ongoing){

            tc_V2G_EVCC_Msg_Timer.start(
            par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

            if(PICS_SECC_CMN_DepartureTime) {

                if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                omit, vc_requestedEnergyTransferModeAC,
                                vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                vc_EVMaxCurrent, vc_EVMinCurrent))));
                } else {

                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                par_CMN_MaxEntriesSAScheduleTuple,
                                vc_requestedEnergyTransferModeAC,
                                vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                vc_EVMaxCurrent, vc_EVMinCurrent))));
                }
            }
            else {

                if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),

```

```

md_CMN_BodyType_001(
md_SECC_AC_ChargeParameterDiscoveryReq_001(
omit, vc_requestedEnergyTransferModeAC,
omit, vc_EAmount, vc_EVMaxVoltage,
vc_EVMaxCurrent, vc_EVMinCurrent)))));
} else {
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_SECC_AC_ChargeParameterDiscoveryReq_001(
                par_CMN_MaxEntriesSAScheduleTuple,
                vc_requestedEnergyTransferModeAC,
                omit, vc_EAmount, vc_EVMaxVoltage,
                vc_EVMaxCurrent, vc_EVMinCurrent)))));
    }
}

alt {
[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(vc_SessionID),
    md_CMN_BodyType_001(
        mdw_SECC_AC_ChargeParameterDiscoveryRes_001(
            finished, md_CMN_AC_EVSEStatus_001(?, ?, ?), ?)))
-> value v_responseMessage {

    v_EVSEprocessing := finished;
    vc_SASchedule_List := v_responseMessage.body.bodyElement.
        chargeParameterDiscoveryRes.sASchedules.
        sAScheduleList;
    vc_PMaxSchedule := v_responseMessage.body.bodyElement.
        chargeParameterDiscoveryRes.sASchedules.
        sAScheduleList.sAScheduleTuple_list[0].
        pMaxSchedule;
    vc_SASchedules := v_responseMessage.body.bodyElement.
        chargeParameterDiscoveryRes.sASchedules;
    vc_SAScheduleTupleId := vc_SASchedule_List.sAScheduleTuple_list[0].
        sAScheduleTupleId;
    vc_EVSENominalVoltage := v_responseMessage.body.bodyElement.
        chargeParameterDiscoveryRes.
        eVSEChargeParameter.aC_EVSEChargeParameter.
        eVSENominalVoltage;
    f_check_PhysicalValue_valueRange(vc_EVSENominalVoltage, v,
        cc_maximum_value_physicalValue_case2,
        fail);
    vc_EVSEMaxCurrent := v_responseMessage.body.bodyElement.
        chargeParameterDiscoveryRes.eVSEChargeParameter.
        aC_EVSEChargeParameter.eVSEMaxCurrent;
    f_check_PhysicalValue_valueRange(vc_EVSEMaxCurrent, a,
        cc_maximum_value_physicalValue_case1,
        fail);
    tc_V2G_EVCC_Msg_Timer.stop;
    tc_V2G_EVCC_Ongoing_Timer.stop;
    if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
        eVSEChargeParameter.aC_EVSEChargeParameter.aC_EVSEStatus.
        eVSENotification != none_) {
        logWarn("The field 'eVSENotification' should be set to 'none_'.");
    }
    if(vc_requestedEnergyTransferModeAC == aC_three_phase_core) {
        v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
            f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent) * 3.0;
    }
    else {
        v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
            f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent);
    }
    f_checkACPMaxScheduleEntries(vc_SASchedule_List, v_pMax, fail);
    if(getverdict == pass) {
        f_checkPMaxSchedulesStructure(vc_SASchedule_List,
            cc_maximum_value_physicalValue_case4,
            PICS_SECC_CMN_MaxEntriesSAScheduleTuple, fail);
    }
    if(getverdict == pass) {
        f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime, vc_SASchedules,
            fail);
    }
    if(getverdict == pass) {
        f_checkSalesTariffStructure(vc_SASchedule_List,
            PICS_SECC_CMN_MaxEntriesSAScheduleTuple, fail);
    }
}

```

```

    }
    if(getverdict == pass) {
        f_checkSalesTariffTimeIntervals(vc_DepartureTime, vc_SASchedules,
                                         fail);
    }
    if(PICS_CMN_CMN_IdentificationMode == pnC) {
        var idDigestValueMap resultIdDigestValueMap :=
        f_calculateIdDigestMapFromBodyType(v_responseMessage.body);
        if(lengthof(resultIdDigestValueMap) > 0) {
            var Signature authSignature := v_responseMessage.header.signature;
            var BodyType chargeParameterResBodyType := v_responseMessage.body;

            isSignatureOk := f_verifySignature(
                chargeParameterResBodyType,
                authSignature,
                vc_contractSignatureCertChain.
                subCertificates.
                x509Certificate_list[0], fail);

            if(isSignatureOk) {
                setverdict(pass, "Valid signature of the sales tariff " &
                               "was received.");
            }
            else {
                setverdict(fail, "Invalid signature of the sales tariff " &
                              "was received.");
            }
        }
    }
    if(getverdict == pass) {
        if(f_compareToPreviousSAScheduleTuple(vc_SASchedule_List,
                                              vc_SAScheduleTupleId)) {
            setverdict(pass, "ChargeParameterDiscoveryRes message was correct. " &
                          "A valid SAScheduleTuple was received after " &
                          "paused session.");
        }
        else {
            setverdict(fail, "Invalid SAScheduleTuple was received after " &
                          "paused session.");
        }
    }
}

[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(vc_SessionID),
    md_CMN_BodyType_001(
        mdw_SECC_AC_ChargeParameterDiscoveryRes_002(
            md_CMN_AC_EVSEStatus_001(?, ?, ?))))
-> value v_responseMessage {

    if(tc_V2G_EVCC_Ongoing_Timer.running==false){
        tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
    }
    if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
        evSEChargeParameter.aC_EVSEChargeParameter.aC_EVSEStatus.
        evSENotification != none_) {
        logWarn("The field 'evSENotification' should be set to 'none_'.");
    }

    [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
                                             "terminated by the SUT.") {
        v_EVSEprocessing := finished;
        break;
    }
    [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                     {chargeParameterDiscoveryRes := ?}) {
        v_EVSEprocessing := finished;
        break;
    }
    [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
                                           "but V2G_Message was expected.") {
        v_EVSEprocessing := finished;
        break;
    }
    [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
                                           "message, but with invalid " &
                                           "data content.") {
        v_EVSEprocessing := finished;
        break;
    }
}

```

```

    [] a_SECC_Timeout(fail) {
        v_EVSEprocessing := finished;
        break;
    }
    [] tc_V2G_EVCC_Ongoing_Timer.timeout{
        v_EVSEprocessing := finished;
        setverdict(fail, "EVSEProcessing = Finished was not sent in time.");
        break;
    }
}
}
}
else {setverdict(inconc, "Requested energy transfer mode is not supported by the SECC.");}
return getverdict;
}

function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_008() runs on SECC_Tester
    return verdicttype {

    var EVSEProcessingType v_EVSEprocessing := ongoing;
    var charstring v_message;
    var boolean isShutdownOSC := false;

    if(f_checkRequestedEnergyTransferMode(vc_requestedEnergyTransferModeAC,
        vc_supportedEnergyTransferMode)) {

        while (v_EVSEprocessing == ongoing){

            tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

            if(PICS_SECC_CMN_DepartureTime) {

                if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                omit, vc_requestedEnergyTransferModeAC,
                                vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                vc_EVMaxCurrent, vc_EVMinCurrent))));
                } else {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                par_CMN_MaxEntriesSAScheduleTuple,
                                vc_requestedEnergyTransferModeAC,
                                vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                vc_EVMaxCurrent, vc_EVMinCurrent))));
                }
            }
            else {

                if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                omit, vc_requestedEnergyTransferModeAC,
                                omit, vc_EAmount, vc_EVMaxVoltage,
                                vc_EVMaxCurrent, vc_EVMinCurrent))));
                } else {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                par_CMN_MaxEntriesSAScheduleTuple,
                                vc_requestedEnergyTransferModeAC,
                                omit, vc_EAmount, vc_EVMaxVoltage,
                                vc_EVMaxCurrent, vc_EVMinCurrent))));
                }
            }
        }

        alt {
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_002(vc_SessionID),
                md_CMN_BodyType_001(
                    mdw_SECC_AC_ChargeParameterDiscoveryRes_001(
                        finished, md_CMN_AC_EVSEStatus_001(?, ?, ?),

```

```

        *))) {

        v_EVSEprocessing := finished;
        setverdict(inconc, "Charge parameter information should not be available.");
    }
    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            mdw_SECC_AC_ChargeParameterDiscoveryRes_002(
                md_CMN_AC_EVSEStatus_001(?,?,?)))))) {

        if(tc_V2G_SECC_Ongoing_Timer.running==false){
            tc_V2G_SECC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Performance_Timeout -
                par_CMN_Transmission_Delay);
        }
    }
    [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
        "terminated by the SUT.") {
        v_EVSEprocessing := finished;
        break;
    }
    [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
        {chargeParameterDiscoveryRes := ?}) {
        v_EVSEprocessing := finished;
        break;
    }
    [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
        "but V2G_Message was expected.") {
        v_EVSEprocessing := finished;
        break;
    }
    [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
        "message, but with invalid " &
        "data content.") {
        v_EVSEprocessing := finished;
        break;
    }
    [] a_SECC_Timeout(fail) {
        v_EVSEprocessing := finished;
        break;
    }
    [] tc_V2G_SECC_Ongoing_Timer.timeout{
        f_SECC_setIsConfirmationFlagDC();
        f_SECC_changeValidFrequencyRange(0,0);
        f_SECC_changeValidDutyCycleRange(100,100);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);
        tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            []
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                "terminated by the SUT.") {
                if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");
                }
                pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                "but TCP connection was " &
                ">>NOT<< terminated by " &
                "the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout();
        }
        v_EVSEprocessing := finished;
        break;
    }
}
}
}
else {setverdict(inconc, "Requested energy transfer mode is not supported " &
    "by the SECC.");}
return getverdict;
}

```

```

function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(in verdicttype v_vct)
    runs on SECC_Tester
    return verdicttype {

    var EVSEProcessingType v_EVSEprocessing := ongoing;
    var V2G_Message v_responseMessage;
    var charstring v_message;
    var boolean isSignatureOk := true;
    var boolean v_receivedOngoing := true;

    if(PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing) {
        v_receivedOngoing := false;
    }

    if(f_checkRequestedEnergyTransfermode(vc_requestedEnergyTransferModeDC,
        vc_supportedEnergyTransferMode)) {

        while (v_EVSEprocessing == ongoing){

            tc_V2G_EVCC_Msg_Timer.start(
                par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

            if(PICS_SECC_CMN_DepartureTime) {

                if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                omit, vc_requestedEnergyTransferModeDC,
                                vc_DepartureTime, vc_DC_EVStatus,
                                vc_EVMaximumCurrentLimit,
                                vc_EVMaximumPowerLimit,
                                vc_EVMaximumVoltageLimit))));
                } else {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                par_CMN_MaxEntriesSAScheduleTuple,
                                vc_requestedEnergyTransferModeDC,
                                vc_DepartureTime, vc_DC_EVStatus,
                                vc_EVMaximumCurrentLimit,
                                vc_EVMaximumPowerLimit,
                                vc_EVMaximumVoltageLimit))));
                }
            }
            else{

                if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                omit, vc_requestedEnergyTransferModeDC, omit,
                                vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                                vc_EVMaximumPowerLimit,
                                vc_EVMaximumVoltageLimit))));
                } else {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                par_CMN_MaxEntriesSAScheduleTuple,
                                vc_requestedEnergyTransferModeDC, omit,
                                vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                                vc_EVMaximumPowerLimit,
                                vc_EVMaximumVoltageLimit))));
                }
            }
        }

        alt {
            [!pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_002(vc_SessionID),
                md_CMN_BodyType_001(
                    mdw_SECC_DC_ChargeParameterDiscoveryRes_001(
                        finished, none, ?, *)))
            )-> value v_responseMessage {

```

```

v_EVSEprocessing := finished;
vc_SASchedule_List := v_responseMessage.body.bodyElement.
    chargeParameterDiscoveryRes.sASchedules.
    sAScheduleList;
vc_SASchedules := v_responseMessage.body.bodyElement.
    chargeParameterDiscoveryRes.sASchedules;
vc_SAScheduleTupleId := vc_SASchedule_List.sAScheduleTuple_list[0].
    sAScheduleTupleID;
vc_PMaxSchedule := v_responseMessage.body.bodyElement.
    chargeParameterDiscoveryRes.sASchedules.sAScheduleList.
    sAScheduleTuple_list[0].pMaxSchedule;
vc_EVSEMaximumCurrentLimit := v_responseMessage.body.bodyElement.
    chargeParameterDiscoveryRes.
    eVSEChargeParameter.
    dC_EVSEChargeParameter.evSEMaximumCurrentLimit;
f_check_PhysicalValue_valueRange(vc_EVSEMaximumCurrentLimit, a,
    cc_maximum_value_physicalValue_case1,
    v_vct);
vc_EVSEMaximumVoltageLimit := v_responseMessage.body.bodyElement.
    chargeParameterDiscoveryRes.
    eVSEChargeParameter.
    dC_EVSEChargeParameter.evSEMaximumVoltageLimit;
f_check_PhysicalValue_valueRange(vc_EVSEMaximumVoltageLimit, v,
    cc_maximum_value_physicalValue_case2,
    v_vct);
vc_EVSEMaximumPowerLimit := v_responseMessage.body.bodyElement.
    chargeParameterDiscoveryRes.
    eVSEChargeParameter.
    dC_EVSEChargeParameter.evSEMaximumPowerLimit;
f_check_PhysicalValue_valueRange(vc_EVSEMaximumPowerLimit, w,
    cc_maximum_value_physicalValue_case4,
    v_vct);
vc_EVSEMinimumVoltageLimit := v_responseMessage.body.bodyElement.
    chargeParameterDiscoveryRes.
    eVSEChargeParameter.
    dC_EVSEChargeParameter.evSEMinimumVoltageLimit;
f_check_PhysicalValue_valueRange(vc_EVSEMinimumVoltageLimit, v,
    cc_maximum_value_physicalValue_case2,
    v_vct);
vc_EVSEMinimumCurrentLimit := v_responseMessage.body.bodyElement.
    chargeParameterDiscoveryRes.
    eVSEChargeParameter.
    dC_EVSEChargeParameter.evSEMinimumCurrentLimit;
f_check_PhysicalValue_valueRange(vc_EVSEMinimumCurrentLimit, a,
    cc_maximum_value_physicalValue_case1,
    v_vct);
vc_EVSEPeakCurrentRipple := v_responseMessage.body.bodyElement.
    chargeParameterDiscoveryRes.eVSEChargeParameter.
    dC_EVSEChargeParameter.evSEPeakCurrentRipple;
f_check_PhysicalValue_valueRange(vc_EVSEPeakCurrentRipple, a,
    cc_maximum_value_physicalValue_case1,
    v_vct);

tc_V2G_EVCC_Msg_Timer.stop;
tc_V2G_EVCC_Ongoing_Timer.stop;
f_checkPMaxSchedulesStructure(vc_SASchedule_List,
    cc_maximum_value_physicalValue_case4,
    PICS_SECC_CMN_MaxEntriesSAScheduleTuple,
    v_vct);
f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime, vc_SASchedules,
    v_vct);

if(getverdict == pass) {
    setverdict(pass, "ChargeParameterDiscoveryRes message " &
        "was correct.");
}
}
[[]pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_002(vc_SessionID),
    md_CMN_BodyType_001(
        mdw_SECC_DC_ChargeParameterDiscoveryRes_002(
            none, ?)))) -> value v_responseMessage {

v_receivedOngoing := true;
if(ispresent(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
    eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
    eVSEIsolationStatus)) {
    if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
        eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.

```



```

        eVSEIsolationStatus != invalid) {
            logWarn("The field 'eVSEIsolationStatus' should be " &
                "set to 'invalid'.");
        }
    }
    if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
        eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
        eVSEStatusCode != eVSE_Ready) {
        logWarn("The field 'eVSEStatusCode' should be set " &
            "to 'eVSE_Ready'.");
    }
    if(tc_V2G_EVCC_Ongoing_Timer.running==false)
    {
        tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
    }
}
[] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
    "terminated by the SUT.") {
    v_EVSEprocessing := finished;
    break; }
[] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
    {chargeParameterDiscoveryRes := ?}) {
    v_EVSEprocessing := finished;
    break;
}
[isError()] a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.") {
    v_EVSEprocessing := finished;
    break;
}
[] a_SECC_Unexpected_Message_Content(v_vct, "Received ChargeParameterDiscoveryRes " &
    "message, but with invalid " &
    "data content.") {
    v_EVSEprocessing := finished;
    break;
}
[] a_SECC_Timeout(v_vct) {
    v_EVSEprocessing := finished;
    break; }
[] tc_V2G_EVCC_Ongoing_Timer.timeout{
    v_EVSEprocessing := finished;
    setverdict(v_vct, "EVSEProcessing = Finished was not sent in time.");
    break;
}
}
}
if(PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing and
    not v_receivedOngoing) {
    setverdict(inconc, "The SUT did not repeat the ChargeParameter" &
        " message sequence.");
}
}
else {setverdict(inconc, "Requested energy transfer mode is not supported by the SECC.");}
return getverdict;
}

function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_002() runs on SECC_Tester
    return verdicttype {
        var charstring v_message;
        var boolean isShutdownOSC := false;

        f_SECC_setIsConfirmationFlagDC();
        f_SECC_changeValidFrequencyRange(0,0);
        f_SECC_changeValidDutyCycleRange(100,100);

        if(f_checkRequestedEnergyTransferMode(vc_requestedEnergyTransferModeDC,
            vc_supportedEnergyTransferMode)) {
            tc_V2G_EVCC_Msg_Timer.start(
                par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq;
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                    md_CMN_MessageHeaderType_001(
                        f_rnd_SessionID(1,429496)),
                    md_CMN_BodyType_001(
                        md_SECC_DC_ChargeParameterDiscoveryReq_001(
                            omit, vc_requestedEnergyTransferModeDC,
                            vc_DepartureTime, vc_DC_EVStatus,
                            vc_EVMaximumCurrentLimit, omit,
                            vc_EVMaximumVoltageLimit))));

```

```

alt {
  []
  a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
    isShutdownOSC := true;
    repeat;
  }
  [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_002(?),
    md_CMN_BodyType_001(
      mdw_SECC_CMN_ChargeParameterDiscoveryRes_001(
        FAILED_UnknownSession)))) {

    tc_V2G_EVCC_Msg_Timer.stop;
    setverdict(pass, "ChargeParameterDiscoveryRes with responseCode " &
      "'FAILED_UnknownSession' is successful.");

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
      e_getPortStatus)) to "local";
    tc_CMN_TCP_Connection_Termination_Timer.start(
      par_CMN_TCP_Connection_Termination_Timeout);
    alt{
      [isShutdownOSC == false]
      a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
      }
      [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
        "terminated by the SUT.") {
        if(isShutdownOSC == false) {
          logWarn("SECC oscillator was not shut down before the " &
            "TCP connection was terminated.");
          pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
          "but TCP connection was " &
          ">>NOT<< terminated by the SUT");
        [] a_SECC_TCP_Connection_Termination_Timeout();
      }
    }
    [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
      "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
      {chargeParameterDiscoveryRes := ?});
    [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
      "but V2G_Message was expected.");
    [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
      "message, but with invalid " &
      "data content.");
    [] a_SECC_Timeout(fail);
  }
}
else {setverdict(inconc, "Requested energy transfer mode is not supported by the SECC.");}
return getverdict;
}

function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_003() runs on SECC_Tester
  return verdicttype {

  var charstring v_message;
  var boolean isShutdownOSC := false;

  f_SECC_setIsConfirmationFlagDC();
  f_SECC_changeValidFrequencyRange(0,0);
  f_SECC_changeValidDutyCycleRange(100,100);

  vc_requestedEnergyTransferModeDC := aC_three_phase_core;
  tc_V2G_EVCC_Msg_Timer.start(
    par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);
  pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(vc_SessionID),
    md_CMN_BodyType_001(
      md_SECC_DC_ChargeParameterDiscoveryReq_001(
        omit, vc_requestedEnergyTransferModeDC, vc_DepartureTime,
        vc_DC_EVStatus, vc_EVMMaximumCurrentLimit,
        omit, vc_EVMMaximumVoltageLimit))));

  alt {
    []
  }
}

```

```

a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
    isShutdownOSC := true;
    repeat;
}
[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_002(vc_SessionID),
    md_CMN_BodyType_001(
        mdw_SECC_CMN_ChargeParameterDiscoveryRes_001(
            FAILED_WrongEnergyTransferMode)))) {

    tc_V2G_EVCC_Msg_Timer.stop;
    setverdict(pass,"ChargeParameterDiscoveryRes with responseCode " &
        "'FAILED_WrongEnergyTransferMode' is successful.");

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_getPortStatus)) to "local";
    tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);
    alt{
        [isShutdownOSC == false]
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP Connection was " &
            "terminated by the SUT.") {
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                    "TCP connection was terminated.");
                pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                "but TCP connection was " &
                ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout();
        }
    }
    [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
        "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
        {chargeParameterDiscoveryRes := ?});
    [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
        "but V2G_Message was expected.");
    [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
        "message, but with invalid " &
        "data content.");
    [] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_004() runs on SECC_Tester
    return verdicttype {

        var charstring v_message;
        var boolean isShutdownOSC := false;

        f_SECC_setIsConfirmationFlagDC();
        f_SECC_changeValidFrequencyRange(0,0);
        f_SECC_changeValidDutyCycleRange(100,100);

        tc_V2G_EVCC_Msg_Timer.start(
            par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_SECC_DC_ChargeParameterDiscoveryReq_002(
                    vc_requestedEnergyTransferModeDC))));

        alt {
            []
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_002(vc_SessionID),
                md_CMN_BodyType_001(
                    mdw_SECC_CMN_ChargeParameterDiscoveryRes_001(

```

```

        FAILED_WrongChargeParameter)))) {

    tc_V2G_EVCC_Msg_Timer.stop;
    setverdict(pass, "ChargeParameterDiscoveryRes with responseCode " &
        "'FAILED_WrongChargeParameter' is successful.");

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_getPortStatus)) to "local";
    tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);
    alt{
        [isShutdownOSC == false]
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
            "terminated by the SUT."){
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                    "TCP connection was terminated.");
                pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                "but TCP connection was " &
                ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout();
        }
    }
    [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
        "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
        {chargeParameterDiscoveryRes := ?});
    [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
        "but V2G_Message was expected.");
    [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
        "message, but with invalid " &
        "data content.");
    [] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005() runs on SECC_Tester
    return verdicttype {

    var EVSEProcessingType v_EVSEprocessing := ongoing;
    var V2G_Message v_responseMessage;
    var charstring v_message;
    var boolean isSignatureOk := true;
    var boolean v_receivedOngoing := true;

    if(PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing) {
        v_receivedOngoing := false;
    }

    if(f_checkRequestedEnergyTransferMode(vc_requestedEnergyTransferModeDC,
        vc_supportedEnergyTransferMode)) {

        while (v_EVSEprocessing == ongoing){

            tc_V2G_EVCC_Msg_Timer.start(
                par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

            if(PICS_SECC_CMN_DepartureTime) {

                if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                omit, vc_requestedEnergyTransferModeDC,
                                vc_DepartureTime, vc_DC_EVStatus,
                                vc_EVMaximumCurrentLimit,
                                vc_EVMaximumPowerLimit,
                                vc_EVMaximumVoltageLimit))));
                } else {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),

```

```

md_CMN_BodyType_001(
md_SECC_DC_ChargeParameterDiscoveryReq_001(
par_CMN_MaxEntriesSAScheduleTuple,
vc_requestedEnergyTransferModeDC,
vc_DepartureTime, vc_DC_EVStatus,
vc_EVMaximumCurrentLimit,
vc_EVMaximumPowerLimit,
vc_EVMaximumVoltageLimit)))));
}
}
else{
if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
md_CMN_MessageHeaderType_001(vc_SessionID),
md_CMN_BodyType_001(
md_SECC_DC_ChargeParameterDiscoveryReq_001(
omit, vc_requestedEnergyTransferModeDC, omit,
vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
vc_EVMaximumPowerLimit,
vc_EVMaximumVoltageLimit)))));
} else {
pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
md_CMN_MessageHeaderType_001(vc_SessionID),
md_CMN_BodyType_001(
md_SECC_DC_ChargeParameterDiscoveryReq_001(
par_CMN_MaxEntriesSAScheduleTuple,
vc_requestedEnergyTransferModeDC, omit,
vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
vc_EVMaximumPowerLimit,
vc_EVMaximumVoltageLimit)))));
}
}
alt {
[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
md_CMN_MessageHeaderType_002(vc_SessionID),
md_CMN_BodyType_001(
mdw_SECC_DC_ChargeParameterDiscoveryRes_001(
finished, none, ?, *)))
-> value v_responseMessage {

v_EVSEprocessing := finished;
vc_SASchedule_List := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.sASchedules.
sAScheduleList;
vc_SASchedules := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.sASchedules;
vc_PMaxSchedule := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.sASchedules.
sAScheduleList.
sAScheduleTuple_list[0].pMaxSchedule;
vc_EVSEMaximumCurrentLimit := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.
eVSEChargeParameter.
dC_EVSEChargeParameter.
eVSEMaximumCurrentLimit;
f_check_PhysicalValue_valueRange(vc_EVSEMaximumCurrentLimit, a,
cc_maximum_value_physicalValue_case1,
fail);
vc_EVSEMaximumVoltageLimit := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.
eVSEChargeParameter.
dC_EVSEChargeParameter.
eVSEMaximumVoltageLimit;
f_check_PhysicalValue_valueRange(vc_EVSEMaximumVoltageLimit, v,
cc_maximum_value_physicalValue_case2,
fail);
vc_EVSEMaximumPowerLimit := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.
eVSEChargeParameter.
dC_EVSEChargeParameter.eVSEMaximumPowerLimit;
f_check_PhysicalValue_valueRange(vc_EVSEMaximumPowerLimit, w,
cc_maximum_value_physicalValue_case4,
fail);
vc_EVSEMinimumVoltageLimit := v_responseMessage.body.bodyElement.
chargeParameterDiscoveryRes.
eVSEChargeParameter.
dC_EVSEChargeParameter.

```

```

        eVSEMinimumVoltageLimit;
    f_check_PhysicalValue_valueRange(vc_EVSEMinimumVoltageLimit, v,
        cc_maximum_value_physicalValue_case2,
        fail);
    vc_EVSEMinimumCurrentLimit := v_responseMessage.body.bodyElement.
        chargeParameterDiscoveryRes.
        eVSEChargeParameter.
        dC_EVSEChargeParameter.
        eVSEMinimumCurrentLimit;
    f_check_PhysicalValue_valueRange(vc_EVSEMinimumCurrentLimit, a,
        cc_maximum_value_physicalValue_case1,
        fail);
    vc_EVSEPeakCurrentRipple := v_responseMessage.body.bodyElement.
        chargeParameterDiscoveryRes.
        eVSEChargeParameter.
        dC_EVSEChargeParameter.
        eVSEPeakCurrentRipple;
    f_check_PhysicalValue_valueRange(vc_EVSEPeakCurrentRipple, a,
        cc_maximum_value_physicalValue_case1,
        fail);

    tc_V2G_EVCC_Msg_Timer.stop;
    tc_V2G_EVCC_Ongoing_Timer.stop;
    f_checkPMaxSchedulesStructure(vc_SASchedule_List,
        cc_maximum_value_physicalValue_case4,
        PICS_SECC_CMN_MaxEntriesSAScheduleTuple,
        fail);
    f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime, vc_SASchedules,
        fail);
    if(getverdict == pass) {
        if(f_compareToPreviousSAScheduleTuple(vc_SASchedule_List,
            vc_SAScheduleTupleId)) {
            setverdict(pass, "ChargeParameterDiscoveryRes message " &
                "was correct. A valid SAScheduleTuple " &
                "was received after paused session.");
        }
        else {
            setverdict(fail, "Invalid SAScheduleTuple was received after " &
                "paused session.");
        }
    }
}
[]pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_002(vc_SessionID),
    md_CMN_BodyType_001(
        mdw_SECC_DC_ChargeParameterDiscoveryRes_002(
            none_, ?))) -> value v_responseMessage {

    if(ispresent(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
        eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
        eVSEIsolationStatus)) {
        if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
            eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
            eVSEIsolationStatus != invalid) {
            logWarn("The field 'eVSEIsolationStatus' should be " &
                "set to 'invalid'.");
        }
    }
    if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
        eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
        eVSEStatusCode != eVSE_Ready) {
        logWarn("The field 'eVSEStatusCode' should be set to 'eVSE_Ready'.");
    }
    if(tc_V2G_EVCC_Ongoing_Timer.running==false)
    {
        tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
    }
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.") {
    v_EVSEprocessing := finished;
    break; }
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {chargeParameterDiscoveryRes := ?}) {
    v_EVSEprocessing := finished;
    break;
}
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.") {
    v_EVSEprocessing := finished;

```

```

        break;
    }
    [] a_SECC_Unexpected_Message_Content(fail, "Received " &
        "ChargeParameterDiscoveryRes " &
        "message, but with invalid " &
        "data content.") {
        v_EVSEprocessing := finished;
        break;
    }
    [] a_SECC_Timeout(fail) {
        v_EVSEprocessing := finished;
        break; }
    [] tc_V2G_EVCC_Ongoing_Timer.timeout{
        v_EVSEprocessing := finished;
        setverdict(fail, "EVSEProcessing = Finished was not sent in time.");
        break;
    }
}
}
if(PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing and
    not v_receivedOngoing) {
    setverdict(inconc, "The SUT did not repeat the ChargeParameter" &
        " message sequence.");
}
}
else {setverdict(inconc, "Requested energy transfer mode is not supported by the SECC.");}
return getverdict;
}

function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(in verdicttype v_vct)
    runs on SECC_Tester
    return verdicttype {

    var EVSEProcessingType v_EVSEprocessing := ongoing;
    var V2G_Message v_responseMessage;
    var charstring v_message;
    var boolean isSignatureOk := true;
    var boolean v_receivedOngoing := true;

    if(PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing) {
        v_receivedOngoing := false;
    }

    if(f_checkRequestedEnergyTransferMode(vc_requestedEnergyTransferModeDC,
        vc_supportedEnergyTransferMode)) {

        while (v_EVSEprocessing == ongoing){

            tc_V2G_EVCC_Msg_Timer.start(
                par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

            if(PICS_SECC_CMN_DepartureTime) {
                if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                omit, vc_requestedEnergyTransferModeDC,
                                vc_DepartureTime, vc_DC_EVStatus,
                                vc_EVMaximumCurrentLimit,
                                vc_EVMaximumPowerLimit,
                                vc_EVMaximumVoltageLimit))));
                } else {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                par_CMN_MaxEntriesSAScheduleTuple,
                                vc_requestedEnergyTransferModeDC,
                                vc_DepartureTime, vc_DC_EVStatus,
                                vc_EVMaximumCurrentLimit,
                                vc_EVMaximumPowerLimit,
                                vc_EVMaximumVoltageLimit))));
                }
            }
            else{

                if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {

```



```

pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(vc_SessionID),
    md_CMN_BodyType_001(
        md_SECC_DC_ChargeParameterDiscoveryReq_001(
            omit, vc_requestedEnergyTransferModeDC, omit,
            vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
            vc_EVMaximumPowerLimit,
            vc_EVMaximumVoltageLimit)))));
} else {
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_SECC_DC_ChargeParameterDiscoveryReq_001(
                par_CMN_MaxEntriesSAScheduleTuple,
                vc_requestedEnergyTransferModeDC, omit,
                vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                vc_EVMaximumPowerLimit,
                vc_EVMaximumVoltageLimit)))));
}
}

alt {
    [pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_002(vc_SessionID),
        md_CMN_BodyType_001(
            mdw_SECC_DC_ChargeParameterDiscoveryRes_001(
                finished, none, ?, ?))))
        -> value v_responseMessage {

        v_EVSEprocessing := finished;
        vc_SASchedule_List := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.sASchedules.
            sAScheduleList;
        vc_SASchedules := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.sASchedules;
        vc_SAScheduleTupleId := vc_SASchedule_List.sAScheduleTuple_list[0].
            sAScheduleTupleId;
        vc_PMaxSchedule := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.sASchedules.sAScheduleList.
            sAScheduleTuple_list[0].pMaxSchedule;
        vc_EVSEMaximumCurrentLimit := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.
            evSEChargeParameter.
            dC_EVSEChargeParameter.evSEMaximumCurrentLimit;
        f_check_PhysicalValue_valueRange(vc_EVSEMaximumCurrentLimit, a,
            cc_maximum_value_physicalValue_case1,
            v_vct);
        vc_EVSEMaximumVoltageLimit := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.
            evSEChargeParameter.
            dC_EVSEChargeParameter.evSEMaximumVoltageLimit;
        f_check_PhysicalValue_valueRange(vc_EVSEMaximumVoltageLimit, v,
            cc_maximum_value_physicalValue_case2,
            v_vct);
        vc_EVSEMaximumPowerLimit := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.
            evSEChargeParameter.
            dC_EVSEChargeParameter.evSEMaximumPowerLimit;
        f_check_PhysicalValue_valueRange(vc_EVSEMaximumPowerLimit, w,
            cc_maximum_value_physicalValue_case4,
            v_vct);
        vc_EVSEMinimumVoltageLimit := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.
            evSEChargeParameter.
            dC_EVSEChargeParameter.evSEMinimumVoltageLimit;
        f_check_PhysicalValue_valueRange(vc_EVSEMinimumVoltageLimit, v,
            cc_maximum_value_physicalValue_case2,
            v_vct);
        vc_EVSEMinimumCurrentLimit := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.
            evSEChargeParameter.
            dC_EVSEChargeParameter.evSEMinimumCurrentLimit;
        f_check_PhysicalValue_valueRange(vc_EVSEMinimumCurrentLimit, a,
            cc_maximum_value_physicalValue_case1,
            v_vct);
        vc_EVSEPeakCurrentRipple := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.evSEChargeParameter.
            dC_EVSEChargeParameter.evSEPeakCurrentRipple;
        f_check_PhysicalValue_valueRange(vc_EVSEPeakCurrentRipple, a,

```

```

        cc_maximum_value_physicalValue_case1,
        v_vct);
tc_V2G_EVCC_Msg_Timer.stop;
tc_V2G_EVCC_Ongoing_Timer.stop;
    f_checkPMaxSchedulesStructure(vc_SASchedule_List,
        cc_maximum_value_physicalValue_case4,
        PICS_SECC_CMN_MaxEntriesSAScheduleTuple,
        v_vct);

    if(getverdict == pass) {
        f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime, vc_SASchedules,
            v_vct);
    }
    if(getverdict == pass) {
        f_checkSalesTariffStructure(vc_SASchedule_List,
            PICS_SECC_CMN_MaxEntriesSAScheduleTuple, v_vct);
    }
    if(getverdict == pass) {
        f_checkSalesTariffTimeIntervals(vc_DepartureTime, vc_SASchedules,
            v_vct);
    }
    if(PICS_CMN_CMN_IdentificationMode == pnC) {
        var IdDigestValueMap resultIdDigestValueMap :=
            f_calculateIdDigestMapFromBodyType(v_responseMessage.body);
        if(lengthof(resultIdDigestValueMap) > 0) {
            var Signature authSignature := v_responseMessage.header.signature;
            var BodyType chargeParameterResBodyType := v_responseMessage.body;

            isSignatureOk := f_verifySignature(
                chargeParameterResBodyType,
                authSignature,
                vc_contractSignatureCertChain.
                subCertificates.
                x509Certificate_list[0], v_vct);

            if(isSignatureOk) {
                setverdict(pass, "Valid signature of the sales tariff " &
                    "was received.");
            }
            else {
                setverdict(v_vct, "Invalid signature of the sales " &
                    "tariff was received.");
            }
        }
    }
    if(getverdict == pass) {
        setverdict(pass, "ChargeParameterDiscoveryRes message " &
            "was correct.");
    }
}
[]pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_002(vc_SessionID),
    md_CMN_BodyType_001(
        mdw_SECC_DC_ChargeParameterDiscoveryRes_002(
            none_, ?))) -> value v_responseMessage {
    v_receivedOngoing := true;
    if(ispresent(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
        eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
        eVSEIsolationStatus)) {
        if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
            eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
            eVSEIsolationStatus != invalid) {
            logWarn("The field 'eVSEIsolationStatus' should be " &
                "set to 'invalid'.");
        }
    }
    if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
        eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
        eVSEStatusCode != eVSE_Ready) {
        logWarn("The field 'eVSEStatusCode' should be set " &
            "to 'eVSE_Ready'.");
    }
    if(tc_V2G_EVCC_Ongoing_Timer.running==false)
    {
        tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
    }
}
[] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
    "terminated by the SUT.") {

```

```

        v_EVSEprocessing := finished;
        break; }
[] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                {chargeParameterDiscoveryRes := ?}) {
    v_EVSEprocessing := finished;
    break;
}
[isError()] a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes received, " &
                                         "but V2G_Message was expected.") {
    v_EVSEprocessing := finished;
    break;
}
[] a_SECC_Unexpected_Message_Content(v_vct, "Received ChargeParameterDiscoveryRes " &
                                           "message, but with invalid " &
                                           "data content.") {
    v_EVSEprocessing := finished;
    break;
}
[] a_SECC_Timeout(v_vct) {
    v_EVSEprocessing := finished;
    break; }
[] tc_V2G_EVCC_Ongoing_Timer.timeout{
    v_EVSEprocessing := finished;
    setverdict(v_vct, "EVSEProcessing = Finished was not sent in time.");
    break;
}
}
}
if(PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing and
   not v_receivedOngoing) {
    setverdict(inconc, "The SUT did not repeat the ChargeParameter" &
                     " message sequence.");
}
}
else {setverdict(inconc, "Requested energy transfer mode is not supported by the SECC.");}
return getverdict;
}

function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007() runs on SECC_Tester
    return verdicttype {

    var EVSEProcessingType v_EVSEprocessing := ongoing;
    var V2G_Message v_responseMessage;
    var charstring v_message;
    var boolean isSignatureOk := true;
    var boolean v_receivedOngoing := true;

    if(PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing) {
        v_receivedOngoing := false;
    }

    if(f_checkRequestedEnergyTransferMode(vc_requestedEnergyTransferModeDC,
                                         vc_supportedEnergyTransferMode)) {

        while (v_EVSEprocessing == ongoing){

            tc_V2G_EVCC_Msg_Timer.start(
                par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

            if(PICS_SECC_CMN_DepartureTime) {

                if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                omit, vc_requestedEnergyTransferModeDC,
                                vc_DepartureTime, vc_DC_EVStatus,
                                vc_EVMaximumCurrentLimit,
                                vc_EVMaximumPowerLimit,
                                vc_EVMaximumVoltageLimit))));
                } else {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                            md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                par_CMN_MaxEntriesSAScheduleTuple,
                                vc_requestedEnergyTransferModeDC,
                                vc_DepartureTime, vc_DC_EVStatus,

```

```

        vc_EVMaximumCurrentLimit,
        vc_EVMaximumPowerLimit,
        vc_EVMaximumVoltageLimit))));
    }
}
else{
    if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_SECC_DC_ChargeParameterDiscoveryReq_001(
                    omit, vc_requestedEnergyTransferModeDC, omit,
                    vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                    vc_EVMaximumPowerLimit,
                    vc_EVMaximumVoltageLimit))));
    } else {
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_SECC_DC_ChargeParameterDiscoveryReq_001(
                    par_CMN_MaxEntriesSAScheduleTuple,
                    vc_requestedEnergyTransferModeDC, omit,
                    vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                    vc_EVMaximumPowerLimit,
                    vc_EVMaximumVoltageLimit))));
    }
}
alt {
    []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_002(vc_SessionID),
        md_CMN_BodyType_001(
            mdw_SECC_DC_ChargeParameterDiscoveryRes_001(
                finished, none, ?, ?)))
        -> value v_responseMessage {

        v_EVSEprocessing := finished;
        vc_SASchedule_List := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.sASchedules.
            sAScheduleList;
        vc_SASchedules := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.sASchedules;
        vc_PMaxSchedule := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.sASchedules.
            sAScheduleList.
            sAScheduleTuple_list[0].pMaxSchedule;
        vc_EVSEMaximumCurrentLimit := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.
            evSEChargeParameter.
            dC_EVSEChargeParameter.
            evSEMaximumCurrentLimit;
        f_check_PhysicalValue_valueRange(vc_EVSEMaximumCurrentLimit, a,
            cc_maximum_value_physicalValue_case1,
            fail);
        vc_EVSEMaximumVoltageLimit := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.
            evSEChargeParameter.
            dC_EVSEChargeParameter.
            evSEMaximumVoltageLimit;
        f_check_PhysicalValue_valueRange(vc_EVSEMaximumVoltageLimit, v,
            cc_maximum_value_physicalValue_case2,
            fail);
        vc_EVSEMaximumPowerLimit := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.
            evSEChargeParameter.
            dC_EVSEChargeParameter.evSEMaximumPowerLimit;
        f_check_PhysicalValue_valueRange(vc_EVSEMaximumPowerLimit, w,
            cc_maximum_value_physicalValue_case4,
            fail);
        vc_EVSEMinimumVoltageLimit := v_responseMessage.body.bodyElement.
            chargeParameterDiscoveryRes.
            evSEChargeParameter.
            dC_EVSEChargeParameter.
            evSEMinimumVoltageLimit;
        f_check_PhysicalValue_valueRange(vc_EVSEMinimumVoltageLimit, v,
            cc_maximum_value_physicalValue_case2,
            fail);
        vc_EVSEMinimumCurrentLimit := v_responseMessage.body.bodyElement.

```

```

        chargeParameterDiscoveryRes.
        eVSEChargeParameter.
        dC_EVSEChargeParameter.
        eVSEMinimumCurrentLimit;
    f_check_PhysicalValue_valueRange(vc_EVSEMinimumCurrentLimit, a,
        cc_maximum_value_physicalValue_case1,
        fail);
    vc_EVSEPeakCurrentRipple := v_responseMessage.body.bodyElement.
        chargeParameterDiscoveryRes.
        eVSEChargeParameter.
        dC_EVSEChargeParameter.
        eVSEPeakCurrentRipple;
    f_check_PhysicalValue_valueRange(vc_EVSEPeakCurrentRipple, a,
        cc_maximum_value_physicalValue_case1,
        fail);

    tc_V2G_EVCC_Msg_Timer.stop;
    tc_V2G_EVCC_Ongoing_Timer.stop;
    f_checkPMaxSchedulesStructure(vc_SASchedule_List,
        cc_maximum_value_physicalValue_case4,
        PICS_SECC_CMN_MaxEntriesSAScheduleTuple,
        fail);

    if(getverdict == pass) {
        f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime, vc_SASchedules,
            fail);
    }
    if(getverdict == pass) {
        f_checkSalesTariffStructure(vc_SASchedule_List,
            PICS_SECC_CMN_MaxEntriesSAScheduleTuple, fail);
    }
    if(getverdict == pass) {
        f_checkSalesTariffTimeIntervals(vc_DepartureTime, vc_SASchedules,
            fail);
    }
    if(PICS_CMN_CMN_IdentificationMode == pnC) {
        var idDigestValueMap resultIdDigestValueMap :=
            f_calculateIdDigestMapFromBodyType(v_responseMessage.body);
        if(lengthof(resultIdDigestValueMap) > 0) {
            var Signature authSignature := v_responseMessage.header.signature_;
            var BodyType chargeParameterResBodyType := v_responseMessage.body;

            isSignatureOk := f_verifySignature(
                chargeParameterResBodyType,
                authSignature,
                vc_contractSignatureCertChain.
                subCertificates.
                x509Certificate_list[0], fail);

            if(isSignatureOk) {
                setverdict(pass, "Valid signature of the sales tariff " &
                    "was received.");
            }
            else {
                setverdict(fail, "Invalid signature of the sales tariff " &
                    "was received.");
            }
        }
    }
    if(getverdict == pass) {
        if(f_compareToPreviousSAScheduleTuple(vc_SASchedule_List,
            vc_SAScheduleTupleId)) {
            setverdict(pass, "ChargeParameterDiscoveryRes message " &
                "was correct. A valid SAScheduleTuple " &
                "was received after paused session.");
        }
        else {
            setverdict(fail, "Invalid SAScheduleTuple was received after " &
                "paused session.");
        }
    }
}

[[]pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_002(vc_SessionID),
    md_CMN_BodyType_001(
        mdw_SECC_DC_ChargeParameterDiscoveryRes_002(
            none_, ?)))) -> value v_responseMessage {

    if(ispresent(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
        eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
        eVSEIsolationStatus)) {

```

```

        if (v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
            eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
            eVSEIsolationStatus != invalid) {
            logWarn("The field 'eVSEIsolationStatus' should be " &
                "set to 'invalid'.");
        }
    }
    if (v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
        eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
        eVSEStatusCode != eVSE_Ready) {
        logWarn("The field 'eVSEStatusCode' should be set to 'eVSE_Ready'.");
    }
    if (tc_V2G_EVCC_Ongoing_Timer.running==false)
    {
        tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
    }
}

[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.") {
    v_EVSEprocessing := finished;
    break; }
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {chargeParameterDiscoveryRes := ?}) {
    v_EVSEprocessing := finished;
    break;
}
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.") {
    v_EVSEprocessing := finished;
    break;
}
[] a_SECC_Unexpected_Message_Content(fail, "Received " &
    "ChargeParameterDiscoveryRes " &
    "message, but with invalid " &
    "data content.") {
    v_EVSEprocessing := finished;
    break;
}
[] a_SECC_Timeout(fail) {
    v_EVSEprocessing := finished;
    break; }
[] tc_V2G_EVCC_Ongoing_Timer.timeout{
    v_EVSEprocessing := finished;
    setverdict(fail, "EVSEProcessing = Finished was not sent in time.");
    break;
}

}

}
if (PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing and
    not v_receivedOngoing) {
    setverdict(inconc, "The SUT did not repeat the ChargeParameter" &
        " message sequence.");
}

}
else {setverdict(inconc, "Requested energy transfer mode is not supported by the SECC.");}
return getverdict;
}

function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_008() runs on SECC_Tester
    return verdicttype {

    var EVSEProcessingType v_EVSEprocessing := ongoing;
    var charstring v_message;
    var boolean isShutdownOSC := false;

    if (f_checkRequestedEnergyTransferMode(vc_requestedEnergyTransferModeDC,
        vc_supportedEnergyTransferMode)) {

        while (v_EVSEprocessing == ongoing){

            tc_V2G_EVCC_Msg_Timer.start(
                par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

            if (PICS_SECC_CMN_DepartureTime) {

                if (not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(

```

```

md_SECC_DC_ChargeParameterDiscoveryReq_001(
omit, vc_requestedEnergyTransferModeDC,
vc_DepartureTime, vc_DC_EVStatus,
vc_EVMaximumCurrentLimit,
vc_EVMaximumPowerLimit,
vc_EVMaximumVoltageLimit)))));
} else {
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
md_CMN_MessageHeaderType_001(vc_SessionID),
md_CMN_BodyType_001(
md_SECC_DC_ChargeParameterDiscoveryReq_001(
par_CMN_MaxEntriesSAScheduleTuple,
vc_requestedEnergyTransferModeDC,
vc_DepartureTime, vc_DC_EVStatus,
vc_EVMaximumCurrentLimit,
vc_EVMaximumPowerLimit,
vc_EVMaximumVoltageLimit)))));
}
}
else{
    if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
md_CMN_MessageHeaderType_001(vc_SessionID),
md_CMN_BodyType_001(
md_SECC_DC_ChargeParameterDiscoveryReq_001(
omit, vc_requestedEnergyTransferModeDC, omit,
vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
vc_EVMaximumPowerLimit,
vc_EVMaximumVoltageLimit)))));
    } else {
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
md_CMN_MessageHeaderType_001(vc_SessionID),
md_CMN_BodyType_001(
md_SECC_DC_ChargeParameterDiscoveryReq_001(
par_CMN_MaxEntriesSAScheduleTuple,
vc_requestedEnergyTransferModeDC, omit,
vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
vc_EVMaximumPowerLimit,
vc_EVMaximumVoltageLimit)))));
    }
}
alt {
    [!pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
md_CMN_MessageHeaderType_002(vc_SessionID),
md_CMN_BodyType_001(
mdw_SECC_DC_ChargeParameterDiscoveryRes_001(
finished, none_, ?, *)))) {

        v_EVSEprocessing := finished;
        if(getverdict == pass) {
            setverdict(pass, "ChargeParameterDiscoveryRes message " &
"was correct.");
        }
    }
    [!pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
md_CMN_MessageHeaderType_002(vc_SessionID),
md_CMN_BodyType_001(
mdw_SECC_DC_ChargeParameterDiscoveryRes_002(
none_, ?)))) {

        if(tc_V2G_EVCC_Ongoing_Timer.running==false)
        {
            tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
        }
    }
    [ a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
"terminated by the SUT.") {
        v_EVSEprocessing := finished;
        break;
    }
    [ a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
{chargeParameterDiscoveryRes := ?}) {
        v_EVSEprocessing := finished;
        break;
    }
    [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
"but V2G_Message was expected.") {

```



```

        v_EVSEprocessing := finished;
        break;
    }
    [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes "
&
        "message, but with invalid " &
        "data content.") {
        v_EVSEprocessing := finished;
        break;
    }
    [] a_SECC_Timeout(fail) {
        v_EVSEprocessing := finished;
        break;
    }
    [] tc_V2G_EVCC_Ongoing_Timer.timeout{
        f_SECC_setIsConfirmationFlagDC();
        f_SECC_changeValidFrequencyRange(0,0);
        f_SECC_changeValidDutyCycleRange(100,100);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            []
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                "terminated by the SUT.") {
                if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");
                    pt_HAL_61851_Internal_Port.clear;
                }
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                "but TCP connection was " &
                ">>NOT<< terminated by " &
                "the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout();
        }
        v_EVSEprocessing := finished;
        break;
    }
}
}
}
else {setverdict(inconc, "Requested energy transfer mode is not supported by the SECC.");}
return getverdict;
}

function f_SECC_CMN_TB_VTB_ChargeParameterDiscovery_001() runs on SECC_Tester
    return verdicttype {

        var charstring v_message;
        var boolean isShutdownOSC := false;

        f_SECC_setIsConfirmationFlagDC();
        f_SECC_changeValidFrequencyRange(0,0);
        f_SECC_changeValidDutyCycleRange(100,100);

        tc_V2G_EVCC_Msg_Timer.start(
        par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
            md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

        alt {
            []
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                md_CMN_MessageHeaderType_001(vc_SessionID),
                md_CMN_BodyType_001(
                mdw_SECC_CMN_ServiceDiscoveryRes_001(

```

```

                                FAILED_SequenceError)))) {
    tc_V2G_EVCC_Msg_Timer.stop;
    setverdict(pass, "ServiceDiscoveryRes with responseCode " &
                "'fAILED_SequenceError' is successful.");

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus)) to "local";
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);
    alt{
        [isShutdownOSC == false]
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                "terminated by the SUT.") {
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");
                pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                "but TCP connection was " &
                                                ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout();
        }
    }
    [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
                                                "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                     {serviceDiscoveryRes := ?});
    [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
                                                "but V2G Message was expected.");
    [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
                                                "message, but with invalid " &
                                                "data content.");
    [] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_ChargeParameterDiscovery_002() runs on SECC_Tester
    return verdicttype {

    timer t_wait;
    var boolean isShutdownOSC := false;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout -
                par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus)) to "local";

    alt {
        [] t_wait.timeout;
    }

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                "terminated by the SUT.") {
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");
                pt_HAL_61851_Internal_Port.clear;
            }
        }
        [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &

```

```

        "but TCP connection was " &
        ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}
return getverdict;
}

function f_checkRequestedEnergyTransfermode(EnergyTransferModeType
    v_requestedEnergyTransferMode,
    SupportedEnergyTransferModeType
    v_supportedEnergyTransferMode)
    return boolean {

    for (var integer i:=0; i<sizeof(v_supportedEnergyTransferMode.
        energyTransferMode_list); i:=i + 1) {
        if (v_supportedEnergyTransferMode.energyTransferMode_list[i] ==
            v_requestedEnergyTransferMode) {
            return true;
        }
    }
    return false;
}

function f_checkPMaxSchedulesStructure(in SAScheduleListType
    vc_SASchedule_List, in float
    cc_maximum_value_physicalValue_case4,
    in boolean v_isMaxEntriesLimit,
    in verdicttype v_vct) {

    for (var integer i:=0; i<sizeof(vc_SASchedule_List.
        sAScheduleTuple_list); i:=i + 1) {
        if (vc_SASchedule_List.sAScheduleTuple_list[i].sAScheduleTupleID < 1 or
            vc_SASchedule_List.sAScheduleTuple_list[i].sAScheduleTupleID > 255) {
            setverdict(v_vct, "The SUT shall use the values 1 to 255 for " &
                "the parameter SAScheduleTupleID.");
            break;
        }
        for (var integer j:=i+1; j<sizeof(vc_SASchedule_List.
            sAScheduleTuple_list); j:=j + 1) {
            if (vc_SASchedule_List.sAScheduleTuple_list[i].sAScheduleTupleID ==
                vc_SASchedule_List.sAScheduleTuple_list[j].sAScheduleTupleID) {
                setverdict(v_vct, "The element 'SAScheduleTupleID' is not unique.");
                break;
            }
        }
        var integer listLength := sizeof(vc_SASchedule_List.sAScheduleTuple_list[i].
            pMaxSchedule.pMaxScheduleEntry_list);

        if (listLength > par_CMN_MaxEntriesSAScheduleTuple and
            PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
            setverdict(v_vct, "The SUT did not have considered 'MaxEntriesSAScheduleTuple' " &
                "within SAScheduleTuple (PMaxScheduleEntryList).");
            break;
        }
        for (var integer k:=0; k<listLength; k:=k + 1) {
            var PhysicalValueType v_pmax := vc_SASchedule_List.sAScheduleTuple_list[i].
                pMaxSchedule.pMaxScheduleEntry_list[k].pMax;
            f_check_PhysicalValue_valueRange(v_pmax, w,
                cc_maximum_value_physicalValue_case4,
                v_vct);
        }
        for (var integer n:=0; n<listLength-1; n:=n + 1) {
            if (ispresent(vc_SASchedule_List.sAScheduleTuple_list[i].pMaxSchedule.
                pMaxScheduleEntry_list[n].timeInterval.
                relativeTimeInterval.duration)) {
                setverdict(v_vct, "The element 'duration' shall only be used for " &
                    "the last interval of the PMaxSchedule.");
                break;
            }
        }
        if (not (ispresent(vc_SASchedule_List.sAScheduleTuple_list[i].pMaxSchedule.
            pMaxScheduleEntry_list[listLength-1].timeInterval.
            relativeTimeInterval.duration))) {
            setverdict(v_vct, "The element 'duration' shall be used for the last " &
                "interval of the PMaxSchedule.");
        }
    }
}

```

```

function f_checkACPMMaxScheduleEntries(in SAScheduleListType vc_SASchedule_List,
    in float v_pMax, in verdicttype v_vct) {
    for (var integer i:=0; i<sizeof(vc_SASchedule_List.
        sAScheduleTuple_list); i:=i + 1) {
        var integer listLength := sizeof(vc_SASchedule_List.sAScheduleTuple_list[i].
            pMaxSchedule.pMaxScheduleEntry_list);
        for (var integer m:=0; m<listLength; m:=m + 1) {
            if(v_pMax < f_CMN_PhysicalValue_GetValue(vc_SASchedule_List.
                sAScheduleTuple_list[i].pMaxSchedule.pMaxScheduleEntry_list[m].pMax)) {
                setverdict(v_vct,"Invalid PMaxSchedule entry. The SUT did not " &
                    "calculated the PMax values based " &
                    "on the parameter EVSENominalVoltage.");
            }
        }
    }
}

function f_checkPMaxSchedulesTimeIntervals(XSDAUX.unsignedInt vc_DepartureTime,
    SASchedules_group vc_SASchedules,
    in verdicttype v_vct){

    var integer v_time24h := 86400;

    for (var integer i:=0; i<sizeof(vc_SASchedules.sAScheduleList); i:=i + 1) {
        var SAScheduleTupleType v_scheduleTuple := vc_SASchedules.sAScheduleList.
            sAScheduleTuple_list[i];
        var integer v_sumPmaxSchedule := 0;

        var integer v_start := v_scheduleTuple.pMaxSchedule.pMaxScheduleEntry_list[0].
            timeInterval.relativeTimeInterval.start ;
        var integer v_stop := v_scheduleTuple.pMaxSchedule.pMaxScheduleEntry_list[
            sizeof(v_scheduleTuple.pMaxSchedule.
                pMaxScheduleEntry_list)-1].timeInterval.
            relativeTimeInterval.start ;

        if(ispresent(v_scheduleTuple.pMaxSchedule.pMaxScheduleEntry_list[
            sizeof(v_scheduleTuple.pMaxSchedule.
                pMaxScheduleEntry_list)-1].timeInterval.
                relativeTimeInterval.duration)) {

            var integer v_duration := v_scheduleTuple.pMaxSchedule.pMaxScheduleEntry_list[
                sizeof(v_scheduleTuple.pMaxSchedule.
                    pMaxScheduleEntry_list)-1].timeInterval.
                    relativeTimeInterval.duration;
            v_sumPmaxSchedule := v_stop - v_start + v_duration;

            if(PICS_SECC_CMN_DepartureTime) {
                if(v_sumPmaxSchedule != vc_DepartureTime) {
                    setverdict(v_vct, "The sum of the individual time intervals " &
                        "(PMaxSchedule) did not match " &
                        "the period of time indicated by the " &
                        "EVCC (DepartureTime).");
                }
            }
            else {
                if(v_sumPmaxSchedule < v_time24h) {
                    setverdict(v_vct, "The sum of the individual time " &
                        "intervals (PMaxSchedule) was " &
                        "smaller than 24h.");
                }
            }
        }
    }
}

function f_compareToPreviousSAScheduleTuple(in SAScheduleListType vc_SASchedule_List,
    SAIDType vc_SAScheduleTupleId)
    return boolean {

    for (var integer i:=0; i<sizeof(vc_SASchedule_List); i:=i + 1) {
        if(vc_SASchedule_List.sAScheduleTuple_list[0].sAScheduleTupleID ==
            vc_SAScheduleTupleId) {
            return true;
        }
    }
    return false;
}

function f_checkSalesTariffStructure(in SAScheduleListType

```

```

        vc_SASchedule_List,
        in boolean v_isMaxEntriesLimit,
        in verdicttype v_vct) {

for (var integer i:=0; i<sizeof(vc_SASchedule_List.
    sAScheduleTuple_list); i:=i + 1) {

    var integer listLength := sizeof(vc_SASchedule_List.sAScheduleTuple_list[i].salesTariff.
        salesTariffEntry_list);

    if(listLength > par_CMN_MaxEntriesSAScheduleTuple and
        PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
        setverdict(v_vct, "The SUT did not have considered 'MaxEntriesSAScheduleTuple' " &
            "within SAScheduleTuple (SalesTariffEntryList).");
        break;
    }
}

function f_checkSalesTariffTimeIntervals(XSDAUX.unsignedInt vc_DepartureTime,
    SASchedules_group vc_SASchedules,
    in verdicttype v_vct){

    var integer v_time24h := 86400;

    for (var integer i:=0; i<sizeof(vc_SASchedules.sAScheduleList); i:=i + 1) {
        var SAScheduleTupleType v_scheduleTuple := vc_SASchedules.sAScheduleList.
            sAScheduleTuple_list[i];

        var integer v_sumSalesTariff := 0;

        var integer v_start := v_scheduleTuple.salesTariff.salesTariffEntry_list[0].
            timeInterval.relativeTimeInterval.start;
        var integer v_stop := v_scheduleTuple.salesTariff.salesTariffEntry_list[
            sizeof(v_scheduleTuple.salesTariff.
                salesTariffEntry_list)-1].timeInterval.
            relativeTimeInterval.start;

        if(ispresent(v_scheduleTuple.salesTariff.salesTariffEntry_list[
            sizeof(v_scheduleTuple.salesTariff.
                salesTariffEntry_list)-1].timeInterval.
            relativeTimeInterval.duration)) {

            var integer v_duration := v_scheduleTuple.salesTariff.salesTariffEntry_list[
                sizeof(v_scheduleTuple.salesTariff.
                    salesTariffEntry_list)-1].timeInterval.
                    relativeTimeInterval.duration;
            v_sumSalesTariff := v_stop - v_start + v_duration;

            if(PICS_SECC_CMN_DepartureTime) {
                if(v_sumSalesTariff != vc_DepartureTime) {
                    setverdict(v_vct, "The sum of the individual time intervals " &
                        "(SalesTariff) did not match " &
                        "the period of time indicated by the " &
                        "EVCC (DepartureTime).");
                }
            }
            else {
                if(v_sumSalesTariff < v_time24h) {
                    setverdict(v_vct, "The sum of the individual time " &
                        "intervals (SalesTariff) was " &
                        "smaller than 24h.");
                }
            }
        }
    }
}

function f_checkEVSEChargeParameter(in PhysicalValueType vc_EVSEMaximumCurrentLimit,
    in PhysicalValueType vc_EVSEMaximumVoltageLimit,
    in PhysicalValueType vc_EVMaximumCurrentLimit,
    in PhysicalValueType vc_EVMaximumVoltageLimit,
    in PhysicalValueType vc_EVTTargetVoltage,
    in verdicttype v_vct) {

    if((f_CMN_PhysicalValue_GetValue(vc_EVMaximumCurrentLimit) >
        f_CMN_PhysicalValue_GetValue(vc_EVSEMaximumCurrentLimit)) or
        (f_CMN_PhysicalValue_GetValue(vc_EVMaximumVoltageLimit) >
        f_CMN_PhysicalValue_GetValue(vc_EVSEMaximumVoltageLimit))) {

```

```

    if (f_CMN_PhysicalValue_GetValue(vc_EVTargetVoltage) >
        f_CMN_PhysicalValue_GetValue(vc_EVSEMaximumVoltageLimit)) {
        setverdict(v_vct, "Invalid EVSE charging limits. Charging is not possible.");
    }
    else {
        logWarn("The EV maximum limits are higher than the EVSE maximum limits. " &
            "The EV should adapt the corresponding parameter to the " &
            "EVSE limits.");
        vc_EVMaximumCurrentLimit := vc_EVSEMaximumCurrentLimit;
        vc_EVMaximumVoltageLimit := vc_EVSEMaximumVoltageLimit;
    }
}
}
}
}

```

I.1.2.8 SECC functions for PowerDelivery

```

module TestBehavior_SECC_PowerDelivery {

    import from ComponentsAndPorts all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from Templates_CMN_V2GMessage all;
    import from Templates_CMN_PowerDelivery all;
    import from Templates_CMN_ServiceDiscovery all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Timer_15118_2 all;
    import from Templates_SECC_ServiceDiscovery all;
    import from DataStructure_HAL_61851 {type IEC_61851_States;}
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Pics_15118_2 all;
    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_ServiceDiscovery all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Pixit_15118 all;
    import from Timer_15118 all;

    private function f_SECC_61851_State_CorDorB(in ChargeProgressType cpt,
        in HAL_61851_Listener listener)
        runs on SECC_Tester {

        var IEC_61851_States v_state;
        // set 61851 state C or B depending on charge progress type
        if (cpt == start_) {
            v_state := PICS_SECC_CMN_EvStateInChargeProgress;
        } else if (cpt == stop_) {
            v_state := B;
        }

        if (cpt != renegotiate_) {
            f_SECC_changeValidStateCondition(v_state);
            f_SECC_setState(v_state, listener);
        }
    }

    altstep a_AC_PowerDeliveryRes_Message(ChargeProgressType c, ResponseCodeType rc)
        runs on SECC_Tester {

        var V2G_Message v_responseMessage;
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_CMN_CMN_PowerDeliveryRes_001(
                    rc, md_CMN_AC_EVSEStatus_002(
                        md_CMN_AC_EVSEStatus_001(?, ?, ?))))))
            -> value v_responseMessage {

            tc_V2G_EVCC_Msg_Timer.stop;
            if (v_responseMessage.body.bodyElement.powerDeliveryRes.evSEStatus.
                ac_EVSEStatus.evSENotification != none_) {
                logWarn("The field 'evSENotification' should be set to 'none_'.");
            }
            setverdict(pass, "PowerDeliveryRes message was correct.");
        }
    }

    altstep a_DC_PowerDeliveryRes_Message(ChargeProgressType c, ResponseCodeType rc)

```

```

runs on SECC_Tester {

var V2G_Message v_responseMessage;
[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(vc_SessionID),
    md_CMN_BodyType_001(
        md_CMN_CMN_PowerDeliveryRes_001(rc,
        md_CMN_DC_EVSEStatus_002(
        md_CMN_DC_EVSEStatus_001(?,?,*,?))))))
-> value v_responseMessage {

    tc_V2G_EVCC_Msg_Timer.stop;
    if(ispresent(v_responseMessage.body.bodyElement.powerDeliveryRes.
        eVSEStatus.dC_EVSEStatus.eVSEIsolationStatus)) {
        if(v_responseMessage.body.bodyElement.powerDeliveryRes.eVSEStatus.
            dC_EVSEStatus.eVSEIsolationStatus != valid) {
            logWarn("The field 'eVSEIsolationStatus' should be set to 'valid'.");
        }
    }
    if(v_responseMessage.body.bodyElement.powerDeliveryRes.eVSEStatus.
        dC_EVSEStatus.eVSEStatusCode != eVSE_Ready) {
        logWarn("The field 'eVSEStatusCode' should be set to 'eVSE_Ready'.");
    }
    if(v_responseMessage.body.bodyElement.powerDeliveryRes.eVSEStatus.
        dC_EVSEStatus.eVSENotification != none) {
        logWarn("The field 'eVSENotification' should be set to 'none'.");
    }
    setverdict(pass, "PowerDeliveryRes message was correct.");
}

}

function f_SECC_AC_TB_VTB_PowerDelivery_001(in ChargeProgressType chargeProgressType,
    in HAL_61851_Listener v_HAL_61851_Listener,
    in verdicttype v_vct)
runs on SECC_Tester return verdicttype {

var charstring v_message;
const integer c_maxProfileEntries := 24;
const integer c_numberProfileEntries := sizeof(vc_PMaxSchedule.pMaxScheduleEntry_list);
var integer v_numberProfileEntries;
if(c_numberProfileEntries > c_maxProfileEntries) {
    v_numberProfileEntries := c_maxProfileEntries;
} else {v_numberProfileEntries := c_numberProfileEntries; }
const integer n := v_numberProfileEntries;
var ProfileEntryType v_profileList[n];

tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);

if(chargeProgressType == start) {
    for (var integer i:=0; i<n; i:=i + 1) {
        v_profileList[i] :=
            md_CMN_CMN_ProfileEntry_001(vc_PMaxSchedule.pMaxScheduleEntry_list[i].
                timeInterval.relativeTimeInterval.start,
                vc_PMaxSchedule.pMaxScheduleEntry_list[i].
                pMax, omit);
    }
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_CMN_AC_PowerDeliveryReq_003(
                chargeProgressType, vc_SAScheduleTupleId,
                md_CMN_CMN_ChargingProfile_001(v_profileList))));
    }
    else {
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_CMN_AC_PowerDeliveryReq_001(
                    chargeProgressType, vc_SAScheduleTupleId))));
    }

f_SECC_61851_State_CorDorB(chargeProgressType, v_HAL_61851_Listener);

alt {
    [] a_AC_PowerDeliveryRes_Message(chargeProgressType, oK);
    [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
        "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
        {powerDeliveryRes := ?});
    [isError()] a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes received, " &

```



```

        "but V2G_Message was expected.");
    [] a_SECC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryRes message, " &
        "but with invalid data content.");
    [] a_SECC_Timeout(v_vct);
}

return getverdict;
}

function f_SECC_AC_TB_VTB_PowerDelivery_002(in ChargeProgressType chargeProgressType,
    in HAL_61851_Listener v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_CMN_CMN_ServiceDiscoveryReq_002(omit)))));

    f_SECC_61851_State_CorDorB(chargeProgressType, v_HAL_61851_Listener);

    alt
    {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                mdw_SECC_CMN_ServiceDiscoveryRes_001(
                    FAILED_SequenceError)))) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass, "ServiceDiscoveryRes with responseCode " &
                "'FAILED_SequenceError' is successful.");

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                e_getPortStatus)) to "local";
            tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
            alt{
                [isShutdownOSC == false]
                a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                    isShutdownOSC := true;
                    repeat;
                }
                [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                    "terminated by the SUT.") {
                    if(isShutdownOSC == false) {
                        logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                    pt_HAL_61851_Internal_Port.clear;
                }
                [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                    "but TCP connection was " &
                    ">>NOT<< terminated by the SUT");
                [] a_SECC_TCP_Connection_Termination_Timeout();
            }
        }
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
            "terminated by the SUT.");
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
            {serviceDiscoveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
            "but V2G_Message was expected.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
            "but with invalid data content.");
        [] a_SECC_Timeout(fail);
    }
}

```

```

    return getverdict;
}

function f_SECC_AC_TB_VTB_PowerDelivery_003(in ChargeProgressType chargeProgressType,
                                             in HAL_61851_Listener v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(
            f_rnd_SessionID(1,429496)),
        md_CMN_BodyType_001(
            md_CMN_AC_PowerDeliveryReq_001(
                chargeProgressType, vc_SAScheduleTupleId))););

    f_SECC_61851_State_CorDorB(chargeProgressType, v_HAL_61851_Listener);

    alt
    {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(?),
            md_CMN_BodyType_001(
                md_CMN_CMN_PowerDeliveryRes_001(
                    FAILED_UnknownSession, ?)))) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass, "PowerDeliveryRes with responseCode " &
                "'FAILED_UnknownSession' is successful.");

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                e_getPortStatus)) to "local";
            tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
            alt{
                [isShutdownOSC == false]
                a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                    isShutdownOSC := true;
                    repeat;
                }
                [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                    "terminated by the SUT.") {
                    if(isShutdownOSC == false) {
                        logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                    pt_HAL_61851_Internal_Port.clear;
                }
                [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                    "but TCP connection was " &
                    ">>NOT<< terminated by the SUT");
                [] a_SECC_TCP_Connection_Termination_Timeout();
            }
        }
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
            "terminated by the SUT.");
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
            {powerDeliveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
            "but V2G_Message was expected.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
            "but with invalid data content.");
        [] a_SECC_Timeout(fail);
    }
    return getverdict;
}

function f_SECC_AC_TB_VTB_PowerDelivery_004(in ChargeProgressType chargeProgressType,

```

```

in HAL_61851_Listener v_HAL_61851_Listener)
runs on SECC_Tester return verdicttype {

var charstring v_message;
const PhysicalValueType power := vc_EVSEMaximumPowerLimit;
var PhysicalValueType mpower := power;
mpower.value_ := 2 * mpower.value_;
var boolean isShutdownOSC := false;

const integer n := 7;
var urn_iso_15118_2_2013_MsgDataTypes.ProfileEntryType v_profileList[n];
v_profileList[0] := md_CMN_CMN_ProfileEntry_001(1400, power, 1);
v_profileList[1] := md_CMN_CMN_ProfileEntry_001(1410, power, 2);
v_profileList[2] := md_CMN_CMN_ProfileEntry_001(1420, power, 3);
v_profileList[3] := md_CMN_CMN_ProfileEntry_001(1430, mpower, 2);
v_profileList[4] := md_CMN_CMN_ProfileEntry_001(1440, power, 1);
v_profileList[5] := md_CMN_CMN_ProfileEntry_001(1450, power, 2);
v_profileList[6] := md_CMN_CMN_ProfileEntry_001(1500, power, 3);

f_SECC_setIsConfirmationFlagDC();
f_SECC_changeValidFrequencyRange(0,0);
f_SECC_changeValidDutyCycleRange(100,100);
deactivate(vc_Default_IEC_61851_ListenerBehavior);

tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(vc_SessionID),
    md_CMN_BodyType_001(
        md_CMN_AC_PowerDeliveryReq_003(
            chargeProgressType, vc_SAScheduleTupleId,
            md_CMN_CMN_ChargingProfile_001(v_profileList))));

f_SECC_61851_State_CorDorB(chargeProgressType, v_HAL_61851_Listener);

alt
{
    []
    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
    }
    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_CMN_CMN_PowerDeliveryRes_001(
                FAILED_ChargingProfileInvalid, ?)))) {

        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(pass, "PowerDeliveryRes with responseCode " &
            "'FAILED_ChargingProfileInvalid' is successful.");

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
            e_getPortStatus)) to "local";
        tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
        alt{
            [isShutdownOSC == false]
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                "terminated by the SUT.") {
                if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");
                }
                pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                "but TCP connection was " &
                ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout();
        }
    }
    [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
        "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
        {powerDeliveryRes := ?});
    [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &

```

```

    "but V2G Message was expected.");
    [] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
    "but with invalid data content.");
    [] a_SECC_Timeout(fail);
  }
  return getverdict;
}

function f_SECC_AC_TB_VTB_PowerDelivery_005(in ChargeProgressType chargeProgressType,
    in HAL_61851_Listener v_HAL_61851_Listener)
    runs on SECC_Tester return verdicttype {

  var charstring v_message;
  var boolean isValid;
  const PhysicalValueType power := vc_EVSEMaximumPowerLimit;
  var boolean isShutdownOSC := false;

  const integer n := 7;
  var urn_iso_15118_2_2013_MsgDataTypes.ProfileEntryType v_profileList[n];
  v_profileList[0] := md_CMN_CMN_ProfileEntry_001(1400, power, 1);
  v_profileList[1] := md_CMN_CMN_ProfileEntry_001(1410, power, 2);
  v_profileList[2] := md_CMN_CMN_ProfileEntry_001(1420, power, 3);
  v_profileList[3] := md_CMN_CMN_ProfileEntry_001(1430, power, 2);
  v_profileList[4] := md_CMN_CMN_ProfileEntry_001(1440, power, 1);
  v_profileList[5] := md_CMN_CMN_ProfileEntry_001(1450, power, 2);
  v_profileList[6] := md_CMN_CMN_ProfileEntry_001(1500, power, 3);

  vc_SAScheduleTupleId := 255;
  isValid := checkInvalidTupleID(vc_SAScheduleTupleId, vc_SASchedule_List);

  f_SECC_setIsConfirmationFlagDC();
  f_SECC_changeValidFrequencyRange(0,0);
  f_SECC_changeValidDutyCycleRange(100,100);
  deactivate(vc_Default_IEC_61851_ListenerBehavior);

  if(isValid){
    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
      md_CMN_MessageHeaderType_001(vc_SessionID),
      md_CMN_BodyType_001(
        md_CMN_AC_PowerDeliveryReq_003(
          chargeProgressType, vc_SAScheduleTupleId,
          md_CMN_CMN_ChargingProfile_001(v_profileList))));

    f_SECC_61851_State_CorDorB(chargeProgressType, v_HAL_61851_Listener);

    alt
    {
      []
      a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
      }
      [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
          md_CMN_CMN_PowerDeliveryRes_001(
            fFAILED_TariffSelectionInvalid, ?)))) {

        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(pass, "PowerDeliveryRes with responseCode " &
          "'fFAILED_TariffSelectionInvalid' is successful.");

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
          e_getPortStatus)) to "local";
        tc_CMN_TCP_Connection_Termination_Timer.start(
          par_CMN_TCP_Connection_Termination_Timeout);
        alt{
          [isShutdownOSC == false]
          a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
          }
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
          "terminated by the SUT.") {
          if(isShutdownOSC == false) {
            logWarn("SECC oscillator was not shut down before the " &
              "TCP connection was terminated.");
            pt_HAL_61851_Internal_Port.clear;
          }
        }
      }
    }
  }
}

```

```

    }
    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                           "but TCP connection was " &
                                           ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
  }
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
                                           "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                  {powerDeliveryRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
                                         "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
                                       "but with invalid data content.");
[] a_SECC_Timeout(fail);
}
}
return getverdict;
}

function f_SECC_AC_TB_VTB_PowerDelivery_006(in HAL_61851_Listener v_HAL_61851_Listener)
  runs on SECC_Tester return verdictType {

  var charstring v_message;
  var boolean isShutdownOSC := false;
  var IEC_61851_States v_state;

  f_SECC_setIsConfirmationFlagDC();
  f_SECC_changeValidFrequencyRange(0,0);
  f_SECC_changeValidDutyCycleRange(100,100);
  deactivate(vc_Default_IEC_61851_ListenerBehavior);

  tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);

  pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(vc_SessionID),
    md_CMN_BodyType_001(
      md_CMN_AC_PowerDeliveryReq_003(renegotiate,
      vc_SAScheduleTupleId, omit))));

  v_state := PICS_SECC_CMN_EvStateInChargeProgress;
  f_SECC_changeValidStateCondition(v_state);
  f_SECC_setState(v_state, v_HAL_61851_Listener);

alt
{
  []
  a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
    isShutdownOSC := true;
    repeat;
  }
  [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(vc_SessionID),
    md_CMN_BodyType_001(
      md_CMN_CMN_PowerDeliveryRes_001(FAILED, ?)))) {

    tc_V2G_EVCC_Msg_Timer.stop;
    setverdict(pass, "PowerDeliveryRes with responseCode 'fAILED' is successful. " &
                    "No PowerDeliveryReq with ChargeProgress " &
                    "set to 'Start' has been received before. ");

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
      e_getPortStatus)) to "local";
    tc_CMN_TCP_Connection_Termination_Timer.start(
      par_CMN_TCP_Connection_Termination_Timeout);
    alt{
      [isShutdownOSC == false]
      a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
      }
      [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                "terminated by the SUT.") {
        if(isShutdownOSC == false) {
          logWarn("SECC oscillator was not shut down before the " &
                  "TCP connection was terminated.");}
        pt_HAL_61851_Internal_Port.clear;
      }
    }
  }
}

```

```

        [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                              "but TCP connection was " &
                                              ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
                                          "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                 {powerDeliveryRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
                                       "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
                                       "but with invalid data content.");
[] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_DC_TB_VTB_PowerDelivery_001(in ChargeProgressType chargeProgressType,
                                             in HAL_61851_Listener v_HAL_61851_Listener,
                                             in boolean v_chargingComplete,
                                             in boolean v_setState,
                                             in verdicttype v_vct)
runs on SECC_Tester
return verdicttype {

var charstring v_message;
const integer c_maxProfileEntries := 24;
const integer c_numberProfileEntries := sizeof(vc_PMaxSchedule.pMaxScheduleEntry_list);
var integer v_numberProfileEntries;
if(c_numberProfileEntries > c_maxProfileEntries) {
    v_numberProfileEntries := c_maxProfileEntries;
} else {v_numberProfileEntries := c_numberProfileEntries; }
const integer n := v_numberProfileEntries;
var ProfileEntryType v_profileList[n];

tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);

if(chargeProgressType == start_) {
    for (var integer i:=0; i<n; i:=i+1) {
        v_profileList[i] :=
            md_CMN_CMN_ProfileEntry_001(vc_PMaxSchedule.pMaxScheduleEntry_list[i].
                                       timeInterval.relativeTimeInterval.start_,
                                       vc_PMaxSchedule.pMaxScheduleEntry_list[i].
                                       pMax, omit);
    }
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_CMN_DC_PowerDeliveryReq_002(chargeProgressType,
            vc_SAScheduleTupleId,
            md_CMN_CMN_ChargingProfile_001(v_profileList),
            md_CMN_DC_EVPowerDeliveryParameter(vc_DC_EVStatus,
            omit, v_chargingComplete))));
    }
else {
    if(chargeProgressType == stop_ and not PICS_SECC_DC_WeldingDetection){
        vc_DC_EVStatus.eVReady := false;
    }

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_CMN_DC_PowerDeliveryReq_002(chargeProgressType,
            vc_SAScheduleTupleId, omit,
            md_CMN_DC_EVPowerDeliveryParameter(vc_DC_EVStatus,
            omit, v_chargingComplete))));
    }

alt
{
    [] a_DC_PowerDeliveryRes_Message(chargeProgressType, oK) {
        if(chargeProgressType == stop_ and v_setState) {
            f_SECC_changeValidStateCondition(B);
            f_SECC_setState(B, v_HAL_61851_Listener);
        }
    }
}
}

```

```

[] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
    {powerDeliveryRes := ?});
[isError()] a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryRes message, " &
    "but with invalid data content.");
[] a_SECC_Timeout(v_vct);
}
return getverdict;
}

function f_SECC_DC_TB_VTB_PowerDelivery_002() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

    alt
    {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                mdw_SECC_CMN_ServiceDiscoveryRes_001(
                    fFAILED_SequenceError)))) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass, "PowerDeliveryRes with responseCode " &
                "'fFAILED_SequenceError' is successful.");

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                e_getPortStatus)) to "local";

            tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
            alt{
                [isShutdownOSC == false]
                a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                    isShutdownOSC := true;
                    repeat;
                }
                [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                    "terminated by the SUT.") {
                    if(isShutdownOSC == false) {
                        logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");
                        pt_HAL_61851_Internal_Port.clear;
                    }
                }
                [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                    "but TCP connection was " &
                    ">>NOT<< terminated by the SUT");
                [] a_SECC_TCP_Connection_Termination_Timeout();
            }
        }
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
            "terminated by the SUT.");
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
            {serviceDiscoveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
            "but V2G_Message was expected.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
            "but with invalid data content.");
        [] a_SECC_Timeout(fail);
    }
}

```



```

    }
    return getverdict;
}

function f_SECC_DC_TB_VTB_PowerDelivery_003(in ChargeProgressType chargeProgressType)
    runs on SECC_Tester return verdictttype {

    var charstring v_message;
    const PhysicalValueType power := vc_EVSEMaximumPowerLimit;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    if(chargeProgressType == stop_ and not PICS_SECC_DC_WeldingDetection){
        vc_DC_EVStatus.evReady := false;
    }

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(
            f_rnd_SessionID(1,429496)),
        md_CMN_BodyType_001(
            md_CMN_DC_PowerDeliveryReq_002(
                chargeProgressType, vc_SAScheduleTupleId, omit,
                md_CMN_DC_EVPowerDeliveryParameter(
                    vc_DC_EVStatus, omit, false))))));

    alt
    {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(?),
            md_CMN_BodyType_001(
                md_CMN_CMN_PowerDeliveryRes_001(
                    FAILED_UnknownSession, ?)))) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"PowerDeliveryRes with responseCode " &
                "'FAILED_UnknownSession' is successful.");

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                e_getPortStatus)) to "local";
            tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
            alt{
                [isShutdownOSC == false]
                a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                    isShutdownOSC := true;
                    repeat;
                }
                [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                    "terminated by the SUT.") {
                    if(isShutdownOSC == false) {
                        logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                    pt_HAL_61851_Internal_Port.clear;
                }
                [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                    "but TCP connection was " &
                    ">>NOT<< terminated by the SUT");
                [] a_SECC_TCP_Connection_Termination_Timeout();
            }
        }
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
            "terminated by the SUT.");
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
            {powerDeliveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
            "but V2G_Message was expected.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
            "but with invalid data content.");
        [] a_SECC_Timeout(fail);
    }
}

```

```

    }
    return getverdict;
}

function f_SECC_DC_TB_VTB_PowerDelivery_004(in ChargeProgressType chargeProgressType)
    runs on SECC_Tester return verdicttype {

    var charstring v_message;
    const PhysicalValueType power := vc_EVSEMaximumPowerLimit;
    const integer n := 7;
    var urn_iso_15118_2_2013_MsgDataTypes.ProfileEntryType v_profileList[n];
    var boolean isShutdownOSC := false;

    var PhysicalValueType mpower := power;
    mpower.value_ := 1 + mpower.value_;

    v_profileList[0] := md_CMN_CMN_ProfileEntry_001(1400, power, 1);
    v_profileList[1] := md_CMN_CMN_ProfileEntry_001(1410, power, 2);
    v_profileList[2] := md_CMN_CMN_ProfileEntry_001(1420, power, 3);
    v_profileList[3] := md_CMN_CMN_ProfileEntry_001(1430, mpower, 2);
    v_profileList[4] := md_CMN_CMN_ProfileEntry_001(1440, power, 1);
    v_profileList[5] := md_CMN_CMN_ProfileEntry_001(1450, power, 2);
    v_profileList[6] := md_CMN_CMN_ProfileEntry_001(1500, power, 3);

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    if(chargeProgressType == stop_ and not PICS_SECC_DC_WeldingDetection){
        vc_DC_EVStatus.evReady := false;
    }

    tc_V2G_EVCC_Msg_Timer.start(par V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            md_CMN_DC_PowerDeliveryReq_002(
                chargeProgressType, vc_SAScheduleTupleId,
                md_CMN_CMN_ChargingProfile_001(v_profileList),
                md_CMN_DC_EVPowerDeliveryParameter(
                    vc_DC_EVStatus, omit, false)))));

    alt
    {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_CMN_CMN_PowerDeliveryRes_001(
                    fFAILED_ChargingProfileInvalid, ?)))) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass, "PowerDeliveryRes with responseCode " &
                "'fFAILED_ChargingProfileInvalid' is successful.");

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                e_getPortStatus)) to "local";
            tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
            alt{
                [isShutdownOSC == false]
                a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                    isShutdownOSC := true;
                    repeat;
                }
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                "terminated by the SUT.") {
                if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
                pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                "but TCP connection was " &

```

```

        ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}
}
[] a_SECC_TCPCConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {powerDeliveryRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
    "but with invalid data content.");
[] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_DC_TB_VTB_PowerDelivery_005(in ChargeProgressType chargeProgressType)
    runs on SECC_Tester return verdictType {

    var charstring v_message;
    var boolean isValid;
    const PhysicalValueType power := vc_EVSEMaximumPowerLimit;
    var boolean isShutdownOSC := false;

    const integer n := 7;
    var urn_iso_15118_2_2013_MsgDataTypes.ProfileEntryType v_profileList[n];
    v_profileList[0] := md_CMN_CMN_ProfileEntry_001(1400, power, 1);
    v_profileList[1] := md_CMN_CMN_ProfileEntry_001(1410, power, 2);
    v_profileList[2] := md_CMN_CMN_ProfileEntry_001(1420, power, 3);
    v_profileList[3] := md_CMN_CMN_ProfileEntry_001(1430, power, 2);
    v_profileList[4] := md_CMN_CMN_ProfileEntry_001(1440, power, 1);
    v_profileList[5] := md_CMN_CMN_ProfileEntry_001(1450, power, 2);
    v_profileList[6] := md_CMN_CMN_ProfileEntry_001(1500, power, 3);

    vc_SAScheduleTupleId := 255;
    isValid := checkInvalidTupleID(vc_SAScheduleTupleId, vc_SASchedule_List);

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    if(chargeProgressType == stop and not PICS_SECC_DC_WeldingDetection){
        vc_DC_EVStatus.evReady := false;
    }

    if(isValid){
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_001(vc_SessionID),
            md_CMN_BodyType_001(
                md_CMN_DC_PowerDeliveryReq_002(
                    chargeProgressType, vc_SAScheduleTupleId,
                    md_CMN_CMN_ChargingProfile_001(v_profileList),
                    md_CMN_DC_EVPowerDeliveryParameter(
                        vc_DC_EVStatus, omit, false))));
            alt
            {
                []
                a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                    isShutdownOSC := true;
                    repeat;
                }
                [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                    md_CMN_MessageHeaderType_001(vc_SessionID),
                    md_CMN_BodyType_001(
                        md_CMN_CMN_PowerDeliveryRes_001(
                            FAILED_TariffSelectionInvalid, ?))) {

                    tc_V2G_EVCC_Msg_Timer.stop;
                    setverdict(pass, "PowerDeliveryRes with responseCode " &
                        "'FAILED_TariffSelectionInvalid' is successful.");

                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                        e_getPortStatus)) to "local";
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                        par_CMN_TCP_Connection_Termination_Timeout);

```

```

alt{
  [isShutdownOSC == false]
  a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
    isShutdownOSC := true;
    repeat;
  }
  [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
    "terminated by the SUT.") {
    if(isShutdownOSC == false) {
      logWarn("SECC oscillator was not shut down before the " &
        "TCP connection was terminated.");}
    pt_HAL_61851_Internal_Port.clear;
  }
  [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
    "but TCP connection was " &
    ">>NOT<< terminated by the SUT");
  [] a_SECC_TCP_Connection_Termination_Timeout();
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
  "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
  {powerDeliveryRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
  "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
  "but with invalid data content.");
[] a_SECC_Timeout(fail);
}
}
return getverdict;
}

```

```

function f_SECC_DC_TB_VTB_PowerDelivery_006() runs on SECC Tester return verdicttype {

```

```

  var charstring v_message;
  var boolean isShutdownOSC := false;

```

```

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

```

```

  tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);

```

```

  pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(vc_SessionID),
    md_CMN_BodyType_001(
      md_CMN_DC_PowerDeliveryReq_002(
        renegotiate,vc_SAScheduleTupleId, omit,
        md_CMN_DC_EVPowerDeliveryParameter(
          vc_DC_EVStatus, omit, false))));

```

```

alt
{

```

```

  []
  a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
    isShutdownOSC := true;
    repeat;
  }
  [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(vc_SessionID),
    md_CMN_BodyType_001(
      md_CMN_CMN_PowerDeliveryRes_001(fAILED, ?)))) {

```

```

    tc_V2G_EVCC_Msg_Timer.stop;
    setverdict(pass,"PowerDeliveryRes with responseCode 'fAILED' is successful. " &
      "No PowerDeliveryReq with ChargeProgress " &
      "set to 'Start' has been received before. ");

```

```

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
      e_getPortStatus)) to "local";

```

```

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);
    alt{
      [isShutdownOSC == false]
      a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
      }
    }

```

```

    }
    [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
        "terminated by the SUT.") {
        if(isShutdownOSC == false) {
            logWarn("SECC oscillator was not shut down before the " &
                "TCP connection was terminated.");}
            pt_HAL_61851_Internal_Port.clear;
        }
    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
        "but TCP connection was " &
        ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}

[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {powerDeliveryRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
    "but with invalid data content.");
[] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_PowerDelivery_001() runs on SECC_Tester
    return verdicttype {

    timer t_wait;
    var boolean isShutdownOSC := false;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout -
        par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_TcpTlsStatusReq_001(
        e_getPortStatus)) to "local";

    alt {
        [] t_wait.timeout;
    }

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IBC_61851_ListenerBehavior);

    tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
            "terminated by the SUT.") {
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                    "TCP connection was terminated.");}
                pt_HAL_61851_Internal_Port.clear;
            }
        [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
            "but TCP connection was " &
            ">>NOT<< terminated by the SUT");
        [] a_SECC_TCP_Connection_Termination_Timeout();
    }
    return getverdict;
}

function checkInvalidTupleID(SAIDType v_id, SAScheduleListType
    vc_SAScheduleTuple_List)
    return boolean {

    for (var integer i:=0; i<sizeof(vc_SAScheduleTuple_List); i:=i + 1) {
        if(v_id == vc_SAScheduleTuple_List.sAScheduleTuple_list[i].sAScheduleTupleID) {
            setverdict(inconc,"Selected tupleID is not invalid.");
        }
    }
}

```

```

        return false;
    }
}
return true;
}
}

```

I.1.2.9 SECC functions for CertificateUpdate

```

module TestBehavior_SECC_CertificateUpdate {

    import from Timer_15118_2 all;
    import from Pics_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from ComponentsAndPorts all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Templates_SECC_CertificateUpdate all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_CMN_V2GMessage all;
    import from Templates_SECC_ServiceDiscovery all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Services_HAL_61851 all;
    import from TestBehavior_EVCC_ServiceDiscovery all;
    import from TestBehavior_SECC_CertificateInstallation all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Services_Security all;
    import from http_www_w3_org_2000_09_xmlsig all;
    import from Templates_CMN_Security all;
    import from LibFunctions_15118_2 all;
    import from urn_iso_15118_2_2013_X509Def all;
    import from DataStructure_PICS_15118_2 all;
    import from Timer_15118 all;

    function f_SECC_CMN_TB_VTB_CertificateUpdate_001(in verdicttype v_vct)
        runs on SECC_Tester
        return verdicttype {

            var V2G_Message v_responseMessage;
            var charstring v_message;
            var boolean isSignatureOk := true;
            var boolean isCertChainOk := false;

            if(f_checkCertificateService(vc_serviceList)) {

                var X509CertificateType v_leafCert :=
                    fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_CONTRACT_LEAF_EXPIRES");
                var X509CertificateType v_subCert2 :=
                    fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB2_VALID");
                var X509CertificateType v_subCert1 :=
                    fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB1_VALID");
                vc_v2gRootCert := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_V2G_ROOT_VALID");

                if(getverdict == pass) {
                    var charstring v_EMAID := fx_getEMAID(v_leafCert);
                    vc_x509IssuerName := vc_v2gRootCert.tBSCertificate.issuer;
                    vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.tBSCertificate.serial_Number));

                    var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
                        md_SECC_CMN_CertificateUpdateReq_001(
                            "certificateUpdateReq",
                            md_CertificateChainType_002(v_leafCert,
                                v_subCert2, v_subCert1),
                            v_EMAID, vc_x509IssuerName,
                            vc_x509SerialNumber));

                    var charstring v_privateKey := fx_loadPrivateKey("CRT_CONTRACT_LEAF_EXPIRES");

                    var Signature aSignature := md_CMN_Signature_001(
                        md_CMN_SignedInfoType_001("certificateUpdateReq",
                            f_generateDigestFromBodyType(
                                myCertificateUpdateBodyType)), v_privateKey);

                    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);

                    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(

```

```

md_CMN_MessageHeaderType_003(
vc_SessionID, aSignature),
myCertificateUpdateBodyType));

alt {
[]pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
md_CMN_MessageHeaderType_002(vc_SessionID),
md_CMN_BodyType_001(
mdw_SECC_CMN_CertificateUpdateRes_001(
m_CMN_sAProvisioningCertificateChain_001(),
m_CMN_contractSignatureCertChain_001(),
v_EMAID)))) -> value v_responseMessage {

tc_V2G_EVCC_Msg_Timer.stop;

vc_sAProvisioningCertChain := v_responseMessage.body.bodyElement.
certificateUpdateRes.
sAProvisioningCertificateChain;
vc_contractSignatureCertChain := v_responseMessage.body.bodyElement.
certificateUpdateRes.
contractSignatureCertChain;
vc_contractSignatureEncryptedPrivateKey := v_responseMessage.body.
bodyElement.
certificateUpdateRes.
contractSignatureEncryptedPrivateKey;
vc_dHpublickey := v_responseMessage.body.bodyElement.
certificateUpdateRes.dHpublickey;

vc_eMAID := v_responseMessage.body.bodyElement.
certificateUpdateRes.eMAID;

if(getverdict == pass) {

var Signature authSignature := v_responseMessage.header.signature;

var BodyType certificateUpdateResBodyType := v_responseMessage.
body;

isSignatureOk := f_verifySignature(
certificateUpdateResBodyType,
authSignature,
vc_sAProvisioningCertChain.
x509Certificate, v_vct);

if(isSignatureOk) {
vc_v2gRootCert :=
fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_V2G_ROOT_VALID");

isCertChainOk := fx_verifyCertChain(vc_sAProvisioningCertChain,
vc_v2gRootCert, "CPS");
}

if(isCertChainOk) {
var boolean result := fx_checkCertificateValidity(
vc_contractSignatureCertChain.
x509Certificate.tBSCertificate.
validity.not_Before,
vc_contractSignatureCertChain.
x509Certificate.tBSCertificate.
validity.not_After);

if(not result) {
setverdict(v_vct,"Invalid Certificate validity was detected.");
}

if(getverdict == pass) {

log("EMAID: " & toString<charstring>(vc_eMAID.base));
result := fx_validateEMAID(vc_eMAID.base);
if(not result) {
setverdict(v_vct,"Invalid EMAID syntax was detected.");
}
else {
setverdict(pass, "CertificateUpdateRes message was correct.");
}

vc_sharedSecret := fx_calculateSharedSecret("CRT_CONTRACT_LEAF_EXPIRES",
vc_dHpublickey);

vc_contractPrivateKey := fx_decryptContractPrivateKey(vc_sharedSecret,
vc_contractSignatureEncryptedPrivateKey);

```



```

//check contract cert private key with given NEW contract cert (public key)
var boolean isValidContractPrivateKey := fx_isPrivateKeyValidForCertificate(
    vc_contractPrivateKey,
    vc_contractSignatureCertChain.
    x509Certificate);

if (not isValidContractPrivateKey) {
    setverdict(v_vct, "Contract private key does not match the " &
        "contract certificate");
} else {
    log("VALID contract private key to contract certificate: ",
        isValidContractPrivateKey);
}
}
}
else {
    setverdict(v_vct, "CertChain of the oEMProvisioningCert is not valid.");
}
}
}
[] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
    {certificateUpdateRes := ?});
[isError()] a_SECC_Unexpected_Message(v_vct, "SupportedAppProtocolRes received, " &
    "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(v_vct, "Received CertificateUpdateRes" &
    " message, but with invalid " &
    "data content.");

[] a_SECC_Timeout(v_vct);
}
}
}
else {setverdict(inconc, "Wrong PIXIT selection. No certificate service has been offered.");}
return getverdict;
}

function f_SECC_CMN_TB_VTB_CertificateUpdate_002() runs on SECC_Tester
    return verdicttype {

var charstring v_message;
var boolean isShutdownOSC := false;

f_SECC_setIsConfirmationFlagDC();
f_SECC_changeValidFrequencyRange(0,0);
f_SECC_changeValidDutyCycleRange(100,100);
deactivate(vc_Default_IEC_61851_ListenerBehavior);

tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);
pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_001(vc_SessionID),
    md_CMN_BodyType_001(
        md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

alt {
    []
    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
    }
    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_001(vc_SessionID),
        md_CMN_BodyType_001(
            mdw_SECC_CMN_ServiceDiscoveryRes_001(
                fFAILED_SequenceError)))) {

        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(pass, "ServiceDiscoveryRes with responseCode " &
            "'fFAILED_SequenceError' is successful.");

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
            e_getPortStatus)) to "local";
        tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
        alt{
            [isShutdownOSC == false]
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
        }
    }
}
}

```

```

    [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                             "terminated by the SUT.") {
        if(isShutdownOSC == false) {
            logWarn("SECC oscillator was not shut down before the " &
                  "TCP connection was terminated.");
            pt_HAL_61851_Internal_Port.clear;
        }
    }
    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                           "but TCP connection was " &
                                           ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
                                           "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                  {serviceDiscoveryRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
                                         "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received CertificateUpdateRes message, " &
                                         "but with invalid data content.");
[] a_SECC_Timeout(fail);
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_CertificateUpdate_003() runs on SECC_Tester
    return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    var X509CertificateType v_leafCert :=
fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_CONTRACT_LEAF_EXPIRES");
    var X509CertificateType v_subCert2 :=
fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB2_VALID");
    var X509CertificateType v_subCert1 :=
fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB1_VALID");
    vc_v2gRootCert := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_V2G_ROOT_VALID");

    if(getverdict == pass) {
        var charstring v_EMAID := fx_getEMAID(v_leafCert);
        vc_x509IssuerName := vc_v2gRootCert.tBSCertificate.issuer;
        vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.tBSCertificate.serial_Number));

        var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
            md_SECC_CMN_CertificateUpdateReq_001(
                "certificateUpdateReq",
                md_CertificateChainType_002(v_leafCert,
                    v_subCert2, v_subCert1),
                v_EMAID, vc_x509IssuerName,
                vc_x509SerialNumber));

        var charstring v_privateKey := fx_loadPrivateKey("CRT_CONTRACT_LEAF_EXPIRES");

        var Signature aSignature := md_CMN_Signature_001(
            md_CMN_SignedInfoType_001("certificateUpdateReq",
                f_generateDigestFromBodyType(
                    myCertificateUpdateBodyType)), v_privateKey);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_003(
                f_rnd_SessionID(1,429496), aSignature),
            myCertificateUpdateBodyType));

        alt {
            []
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
        }
        [pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(

```

```

md_CMN_MessageHeaderType_002(?),
md_CMN_BodyType_001(
mdw_SECC_CMN_CertificateUpdateRes_002(
FAILED_UnknownSession)))) {
tc_V2G_EVCC_Msg_Timer.stop;
setverdict(pass,"CertificateUpdateRes with responseCode " &
"'FAILED_UnknownSession' is successful.");

pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
e_getPortStatus)) to "local";
tc_CMN_TCP_Connection_Termination_Timer.start(
par_CMN_TCP_Connection_Termination_Timeout);
alt{
[isShutdownOSC == false]
a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
isShutdownOSC := true;
repeat;
}
[] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
"terminated by the SUT.") {
if(isShutdownOSC == false) {
logWarn("SECC oscillator was not shut down before the " &
"TCP connection was terminated.");}
pt_HAL_61851_Internal_Port.clear;
}
[] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
"but TCP connection was " &
">>NOT<< terminated by the SUT");
[] a_SECC_TCP_Connection_Termination_Timeout();
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
"terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
{certificateUpdateRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
"but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received CertificateUpdateRes " &
"message, but with invalid " &
"data content.");
[] a_SECC_Timeout(fail);
}
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_CertificateUpdate_004() runs on SECC_Tester
return verdicttype {

timer t_wait;
var boolean isShutdownOSC := false;

// Wait until sequence timeout in SUT should expire
t_wait.start(par_V2G_SECC_Sequence_Timeout -
par_CMN_Transmission_Delay);
pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
e_getPortStatus)) to "local";

alt {
[] t_wait.timeout;
}

f_SECC_setIsConfirmationFlagDC();
f_SECC_changeValidFrequencyRange(0,0);
f_SECC_changeValidDutyCycleRange(100,100);
deactivate(vc_Default_IEC_61851_ListenerBehavior);

tc_CMN_TCP_Connection_Termination_Timer.start(
par_CMN_TCP_Connection_Termination_Timeout);

alt {
[]
a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
isShutdownOSC := true;
repeat;
}
[] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
"terminated by the SUT.") {
if(isShutdownOSC == false) {

```

```

        logWarn("SECC oscillator was not shut down before the " &
                "TCP connection was terminated.");}
        pt_HAL_61851_Internal_Port.clear;
    }
    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
            "but TCP connection was " &
            ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_CertificateUpdate_005() runs on SECC_Tester
    return verdicttype {

var charstring v_message;
var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

var X509CertificateType v_leafCert :=
fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_CONTRACT_LEAF_EXPIRES");
var X509CertificateType v_subCert2 :=
fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB2_VALID");
var X509CertificateType v_subCert1 :=
fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB1_VALID");
vc_v2gRootCert := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_V2G_ROOT_VALID");

if(getverdict == pass) {
    var charstring v_EMAID := fx_getEMAID(v_leafCert);
    vc_x509IssuerName := vc_v2gRootCert.tBSCertificate.issuer;
    vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.tBSCertificate.serial_Number));

    var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
        md_SECC_CMN_CertificateUpdateReq_001(
            "certificateUpdateReq",
            md_CertificateChainType_002(v_leafCert,
                v_subCert2, v_subCert1),
            v_EMAID, vc_x509IssuerName,
            vc_x509SerialNumber));

    // load wrong private key
    var charstring v_privateKey := fx_loadPrivateKey("CRT_V2G_ROOT_VALID");

    var Signature aSignature := md_CMN_Signature_001(
        md_CMN_SignedInfoType_001("certificateUpdateReq",
            f_generateDigestFromBodyType(
                myCertificateUpdateBodyType)), v_privateKey);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_003(
            vc_SessionID, aSignature),
        myCertificateUpdateBodyType));

    alt {
        []
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
    }
    []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_002(vc_SessionID),
        md_CMN_BodyType_001(
            mdw_SECC_CMN_CertificateUpdateRes_002(
                FAILED_SignatureError)))) {

        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(pass, "CertificateUpdateRes with responseCode " &
            "'FAILED_SignatureError' is successful.");

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
            e_getPortStatus)) to "local";
        tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
        alt{

```

```

[isShutdownOSC == false]
a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
    isShutdownOSC := true;
    repeat;
}
[] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
    "terminated by the SUT.") {
    if(isShutdownOSC == false) {
        logWarn("SECC oscillator was not shut down before the " &
            "TCP connection was terminated.");}
    pt_HAL_61851_Internal_Port.clear;
}
[] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
    "but TCP connection was " &
    ">>NOT<< terminated by the SUT");
[] a_SECC_TCP_Connection_Termination_Timeout();
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
    "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
    {certificateUpdateRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
    "but V2G Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received CertificateUpdateRes " &
    "message, but with invalid " &
    "data content.");
[] a_SECC_Timeout(fail);
}
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_CertificateUpdate_006() runs on SECC_Tester
    return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    var X509CertificateType v_leafCert :=
    fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_CONTRACT_LEAF_EXPIRES");
    var X509CertificateType v_subCert2 :=
    fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB2_VALID");
    var X509CertificateType v_subCert1 :=
    fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB1_VALID");
    vc_v2gRootCert := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_V2G_ROOT_VALID");

    if(getverdict == pass) {
        var charstring v_EMAID := fx_getEMAID(v_leafCert);
        vc_x509IssuerName := vc_v2gRootCert.tBSCertificate.issuer;
        vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.tBSCertificate.serial_Number));

        var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
            md_SECC_CMN_CertificateUpdateReq_001(
                "certificateUpdateReq",
                md_CertificateChainType_002(v_leafCert,
                    v_subCert2, v_subCert1),
                v_EMAID, vc_x509IssuerName,
                vc_x509SerialNumber));

        var charstring v_privateKey := fx_loadPrivateKey("CRT_CONTRACT_LEAF_EXPIRES");

        var Signature aSignature := md_CMN_Signature_001(
            md_CMN_SignedInfoType_001("certificateUpdateReq",
                f_generateDigestFromBodyType(
                    myCertificateUpdateBodyType)), v_privateKey);

        // change id after signature generation
        myCertificateUpdateBodyType.bodyElement.
        certificateUpdateReq.id := "contentChanged";

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(

```

```

md_CMN_MessageHeaderType_003(
    vc_SessionID, aSignature),
myCertificateUpdateBodyType));

alt {
    []
    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
    }
    []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_002(vc_SessionID),
        md_CMN_BodyType_001(
            mdw_SECC_CMN_CertificateUpdateRes_002(
                FAILED_SignatureError)))) {

        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(pass,"CertificateUpdateRes with responseCode " &
            "'FAILED_SignatureError' is successful.");

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
            e_getPortStatus)) to "local";
        tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
        alt{
            [isShutdownOSC == false]
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                "terminated by the SUT.") {
                if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");
                    pt_HAL_61851_Internal_Port.clear;
                }
                [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                    "but TCP connection was " &
                    ">>NOT<< terminated by the SUT");
                [] a_SECC_TCP_Connection_Termination_Timeout();
            }
        }
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
            "terminated by the SUT.");
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
            {certificateUpdateRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
            "but V2G Message was expected.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateUpdateRes " &
            "message, but with invalid " &
            "data content.");
        [] a_SECC_Timeout(fail);
    }
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_CertificateUpdate_007(charstring v_config) runs on SECC_Tester
return verdicttype {
    var charstring v_message;
    var PkiType v_wrongPki := PKI_2;
    var boolean isShutdownOSC := false;
    var X509CertificateType v_leafCert;
    var X509CertificateType v_subCert1;
    var X509CertificateType v_subCert2;
    var charstring v_config1 := "WrongSubCertOrder";
    var charstring v_config2 := "WrongPKILeafCert";
    var charstring v_config3 := "WrongPKISub2Cert";
    var charstring v_config4 := "WrongPKISub1Cert";
    var charstring v_config5 := "WrongPKICertChain";

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    if(v_config == v_config1) {
        v_leafCert := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_CONTRACT_LEAF_EXPIRES");
    }
}

```

```

    v_subCert2 := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB1_VALID");
    v_subCert1 := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB2_VALID");
} else if(v_config == v_config2) {
    v_leafCert := fx_loadCertificate(v_wrongPki, "CRT_CONTRACT_LEAF_EXPIRES");
    v_subCert2 := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB2_VALID");
    v_subCert1 := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB1_VALID");
} else if(v_config == v_config3) {
    v_leafCert := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_CONTRACT_LEAF_EXPIRES");
    v_subCert2 := fx_loadCertificate(v_wrongPki, "CRT_MO_SUB2_VALID");
    v_subCert1 := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB1_VALID");
} else if(v_config == v_config4) {
    v_leafCert := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_CONTRACT_LEAF_EXPIRES");
    v_subCert2 := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB2_VALID");
    v_subCert1 := fx_loadCertificate(v_wrongPki, "CRT_MO_SUB1_VALID");
} else if(v_config == v_config5) {
    v_leafCert := fx_loadCertificate(v_wrongPki, "CRT_CONTRACT_LEAF_EXPIRES");
    v_subCert2 := fx_loadCertificate(v_wrongPki, "CRT_MO_SUB2_VALID");
    v_subCert1 := fx_loadCertificate(v_wrongPki, "CRT_MO_SUB1_VALID");
} else {setverdict(inconc, "Unknown configuration was detected.");}
vc_v2gRootCert := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_V2G_ROOT_VALID");

if(getverdict == pass) {
    var charstring v_EMAID := fx_getEMAID(v_leafCert);
    vc_x509IssuerName := vc_v2gRootCert.tBSCertificate.issuer;
    vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.tBSCertificate.serial_Number));

    var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
        md_SECC_CMN_CertificateUpdateReq_001(
            "certificateUpdateReq",
            md_CertificateChainType_002(v_leafCert,
                v_subCert2, v_subCert1,
                v_EMAID, vc_x509IssuerName,
                vc_x509SerialNumber));

    var charstring v_privateKey := fx_loadPrivateKey("CRT_CONTRACT_LEAF_EXPIRES");

    var Signature aSignature := md_CMN_Signature_001(
        md_CMN_SignedInfoType_001("certificateUpdateReq",
            f_generateDigestFromBodyType(
                myCertificateUpdateBodyType)), v_privateKey);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_003(
            vc_SessionID, aSignature),
        myCertificateUpdateBodyType));

alt {
    []
    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
    }
    []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_002(vc_SessionID),
        md_CMN_BodyType_001(
            mdw_SECC_CMN_CertificateUpdateRes_002(
                FAILED_CertChainError)))) {
        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(pass, "CertificateUpdateRes with responseCode " &
            "'FAILED_CertChainError' is successful.");

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
            e_getPortStatus)) to "local";
        tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
        alt{
            [isShutdownOSC == false]
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                "terminated by the SUT.") {
                if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
                    pt_HAL_61851_Internal_Port.clear;

```



```

    }
    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                           "but TCP connection was " &
                                           ">>NOT<< terminated by the SUT");
    [] a_SECC_TCP_Connection_Termination_Timeout();
  }
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
                                           "terminated by the SUT.");
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                  {certificateUpdateRes := ?});
[isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
                                         "but V2G_Message was expected.");
[] a_SECC_Unexpected_Message_Content(fail, "Received CertificateUpdateRes " &
                                         "message, but with invalid " &
                                         "data content.");

[] a_SECC_Timeout(fail);
}
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_CertificateUpdate_008(charstring v_EMAID_Unknown)
    runs on SECC_Tester
    return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var ResponseCodeType v_responseCodeReceived;
    var V2G_Message v_responseMessage;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

    var X509CertificateType v_leafCert :=
    fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_CONTRACT_LEAF_EXPIRES");
    var X509CertificateType v_subCert2 :=
    fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB2_VALID");
    var X509CertificateType v_subCert1 :=
    fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB1_VALID");
    vc_v2gRootCert := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_V2G_ROOT_VALID");
    var charstring v_EMAID := fx_getEMAID(v_leafCert);

    if(v_EMAID != v_EMAID_Unknown) {

        v_EMAID := v_EMAID_Unknown;
        vc_x509IssuerName := vc_v2gRootCert.tbSCertificate.issuer;
        vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.tbSCertificate.serial_Number));

        var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
            md_SECC_CMN_CertificateUpdateReq_001(
                "certificateUpdateReq",
                md_CertificateChainType_002(v_leafCert,
                    v_subCert2, v_subCert1),
                v_EMAID, vc_x509IssuerName,
                vc_x509SerialNumber));

        var charstring v_privateKey := fx_loadPrivateKey("CRT_CONTRACT_LEAF_EXPIRES");

        var Signature aSignature := md_CMN_Signature_001(
            md_CMN_SignedInfoType_001("certificateUpdateReq",
                f_generateDigestFromBodyType(
                    myCertificateUpdateBodyType)), v_privateKey);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
            md_CMN_MessageHeaderType_003(
                vc_SessionID, aSignature),
            myCertificateUpdateBodyType));

        alt {
            []
            a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                isShutdownOSC := true;
                repeat;
            }
        }
    }
}

```

```

[]pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_002(vc_SessionID),
    md_CMN_BodyType_001(
        mdw_SECC_CMN_CertificateUpdateRes_002(
            m_CMN_failedResponseCodeList_002()))))
-> value v_responseMessage {

    tc_V2G_EVCC_Msg_Timer.stop;
    v_responseCodeReceived := v_responseMessage.body.bodyElement.
        certificateUpdateRes.responseCode;
    setverdict(pass, "CertificateUpdateRes with responseCode '" &
        toString<ResponseCodeType>(v_responseCodeReceived) &
        "' is successful.");

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
        e_getPortStatus)) to "local";
    tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);
    alt{
        [isShutdownOSC == false]
        a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
            isShutdownOSC := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
            "terminated by the SUT.") {
            if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                    "TCP connection was terminated.");
                pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                "but TCP connection was " &
                ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout();
        }
    }
    [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
        "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
        {certificateUpdateRes := ?});
    [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
        "but V2G_Message was expected.");
    [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateUpdateRes " &
        "message, but with invalid " &
        "data content.");
    [] a_SECC_Timeout(fail);
}
} else {
    setverdict(inconc, "Unknown EMAID was valid for the Contract Leaf Certificate.");
}
return getverdict;
}

function f_SECC_CMN_TB_VTB_CertificateUpdate_009(charstring v_certName)
    runs on SECC_Tester
    return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var ResponseCodeType v_responseCodeReceived;
    var V2G_Message v_responseMessage;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);
    var X509CertificateType v_leafCertUnused :=
    fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_CONTRACT_LEAF_EXPIRES");
    var charstring v_EMAID := fx_getEMAID(v_leafCertUnused);
    var X509CertificateType v_leafCert :=
    fx_loadCertificate(PICS_CMN_CMN_PkiName, v_certName);
    var X509CertificateType v_subCert2 :=
    fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB2_VALID");
    var X509CertificateType v_subCert1 :=
    fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB1_VALID");
    vc_v2gRootCert := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_V2G_ROOT_VALID");
    var charstring v_EMAID_Unknown := fx_getEMAID(v_leafCert);

    if(v_EMAID != v_EMAID_Unknown) {

```

```

vc_x509IssuerName := vc_v2gRootCert.tBSCertificate.issuer;
vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.tBSCertificate.serial_Number));

var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
    md_SECC_CMN_CertificateUpdateReq_001(
        "certificateUpdateReq",
        md_CertificateChainType_002(v_leafCert,
            v_subCert2, v_subCert1),
        v_EMAID, vc_x509IssuerName,
        vc_x509SerialNumber));

var charstring v_privateKey := fx_loadPrivateKey(v_certName);

var Signature aSignature := md_CMN_Signature_001(
    md_CMN_SignedInfoType_001("certificateUpdateReq",
        f_generateDigestFromBodyType(
            myCertificateUpdateBodyType)), v_privateKey);

tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);

pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
    md_CMN_MessageHeaderType_003(
        vc_SessionID, aSignature),
    myCertificateUpdateBodyType));

alt {
    []
    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
    }
    []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_002(vc_SessionID),
        md_CMN_BodyType_001(
            mdw_SECC_CMN_CertificateUpdateRes_002(
                m_CMN_failedResponseCodeList_002()))))
        -> value v_responseMessage {
            tc_V2G_EVCC_Msg_Timer.stop;
            v_responseCodeReceived := v_responseMessage.body.bodyElement.
                certificateUpdateRes.responseCode;
            setverdict(pass, "CertificateUpdateRes with responseCode '" &
                toString<ResponseCodeType>(v_responseCodeReceived) &
                "' is successful.");

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
                e_getPortStatus)) to "local";
            tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
            alt{
                [isShutdownOSC == false]
                a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
                    isShutdownOSC := true;
                    repeat;
                }
                [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                    "terminated by the SUT.") {
                    if(isShutdownOSC == false) {
                        logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");
                        pt_HAL_61851_Internal_Port.clear;
                    }
                }
                [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                    "but TCP connection was " &
                    ">>NOT<< terminated by the SUT");
                [] a_SECC_TCP_Connection_Termination_Timeout();
            }
            [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was misleadingly " &
                "terminated by the SUT.");
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                {certificateUpdateRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "SupportedAppProtocolRes received, " &
                "but V2G Message was expected.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateUpdateRes " &
                "message, but with invalid " &
                "data content.");
            [] a_SECC_Timeout(fail);
        }
}

```

```

    } else {
        setverdict(inconc, "Unknown EMAID was valid for the Contract Leaf Certificate.");
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_CertificateUpdate_010(in charstring v_certName,
                                                in template ResponseCodeType
                                                v_responseCode)
    runs on SECC_Tester
    return verdicttype {

var charstring v_message;
var boolean isShutdownOSC := false;
var ResponseCodeType v_responseCodeReceived;
var V2G_Message v_responseMessage;

    f_SECC_setIsConfirmationFlagDC();
    f_SECC_changeValidFrequencyRange(0,0);
    f_SECC_changeValidDutyCycleRange(100,100);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

var X509CertificateType v_leafCert :=
fx_loadCertificate(PICS_CMN_CMN_PkiName, v_certName);
var X509CertificateType v_subCert2 :=
fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB2_VALID");
var X509CertificateType v_subCert1 :=
fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_MO_SUB1_VALID");
vc_v2gRootCert := fx_loadCertificate(PICS_CMN_CMN_PkiName, "CRT_V2G_ROOT_VALID");

if(getverdict == pass) {
    var charstring v_EMAID := fx_getEMAID(v_leafCert);
    vc_x509IssuerName := vc_v2gRootCert.tBSCertificate.issuer;
    vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.tBSCertificate.serial_Number));

    var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
                                                md_SECC_CMN_CertificateUpdateReq_001(
                                                    "certificateUpdateReq",
                                                    md_CertificateChainType_002(v_leafCert,
                                                        v_subCert2, v_subCert1),
                                                    v_EMAID, vc_x509IssuerName,
                                                    vc_x509SerialNumber));

    var charstring v_privateKey := fx_loadPrivateKey(v_certName);

    var Signature aSignature := md_CMN_Signature_001(
        md_CMN_SignedInfoType_001("certificateUpdateReq",
        f_generateDigestFromBodyType(
        myCertificateUpdateBodyType)), v_privateKey);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_003(
        vc_SessionID, aSignature),
        myCertificateUpdateBodyType));

alt
[
    a_CMN_shutdownOscillator(pt_HAL_61851_Internal_Port){
        isShutdownOSC := true;
        repeat;
    }
    []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
        md_CMN_MessageHeaderType_002(vc_SessionID),
        md_CMN_BodyType_001(
        mdw_SECC_CMN_CertificateUpdateRes_002(
        v_responseCode)))
        -> value v_responseMessage {

        tc_V2G_EVCC_Msg_Timer.stop;
        v_responseCodeReceived := v_responseMessage.body.bodyElement.
            certificateUpdateRes.responseCode;
        setverdict(pass, "CertificateUpdateRes with responseCode '" &
            toString<ResponseCodeType>(v_responseCodeReceived) &
            "' is successful.");

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_tcpTlsStatusReq_001(
            e_getPortStatus)) to "local";
        tc_CMN_TCP_Connection_Termination_Timer.start(

```