INTERNATIONAL STANDARD

ISO 13400-2

Second edition
2019-12

Road vehicles — Diagnostic communication over Internet Protocol (DoIP) —

Part 2: Transport protocol and network layer services

Véhicules routiers — Communication de diagnostic au travers du protocole internet (DoIP) —

Partie 2: Protocole de transport et services de la couche réseau

Reference number
ISO 13400-2:2019(E)

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

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 of 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 www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 31, *Data communication*.

This second edition cancels and replaces the first edition (ISO 13400-2:2012), which has been technically revised.

The main changes compared to the previous edition are as follows:

— addition of TLS (Transport Layer Security);

— major restructuring of document content.

A list of all parts in the ISO 13400 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

Vehicle diagnostic communication has been developed starting with the introduction of the first legislated emissions-related diagnostics and has evolved over the years, now covering various use cases ranging from emission-related diagnostics to vehicle-manufacturer-specific applications like calibration or electronic component software updates.

With the introduction of new in-vehicle network communication technologies, the interface between the vehicle's servers and the client DoIP entity has been adapted several times to address the specific characteristics of each new network communication technology requiring optimized data link layer definitions and transport protocol developments in order to make the new in-vehicle networks usable for diagnostic communication.

With increasing memory size of servers, the demand to update this increasing amount of software and an increasing number of functions provided by these control units, technology of the connecting network and buses has been driven to a level of complexity and speed similar to computer networks. Various applications (x-by-wire, infotainment) require high band-width and real-time networks (like FlexRay, MOST), which cannot be adapted to provide the direct interface to a vehicle. This requires gateways to route and convert messages between the in-vehicle networks and the vehicle interface to client DoIP entity.

All parts of ISO 13400 are applicable to vehicle diagnostic systems implemented on an IP communication network.

The ISO 13400 series has been established in order to define common requirements for vehicle diagnostic systems implemented on an IP communication link.

Although primarily intended for diagnostic systems, ISO 13400 has been developed to also meet requirements from other IP-based systems needing a transport protocol and network layer services.

The intent of the ISO 13400 series is to describe a standardized vehicle interface which

— separates in-vehicle network technology from the client DoIP entity vehicle interface requirements to allow for a long-term stable external vehicle communication interface,

— utilizes existing industry standards to define a long-term stable state-of-the-art communication standard usable for legislated diagnostic communication as well as for manufacturer-specific use cases,

— can easily be adapted to new physical and data link layers, including wired and wireless connections, by using existing adaptation layers, and

— allows connections of vehicle-internal and vehicle-external DoIP entities.

To achieve this, it is based on the Open Systems Interconnection (OSI) Basic Reference Model specified in ISO/IEC 7498-1 and ISO/IEC 10731[1], which structures communication systems into seven layers.

Figure 1 illustrates an overview of communication frameworks beyond the scope of this document including related standards:

— Vehicle diagnostic communication framework, which is composed of ISO 14229-1[3], ISO 14229-2[4], and ISO 14229-5[5].

— Presentation layer standards, for example vehicle manufacturer- (VM-) specific or ISO 22901 ODX[6].

— OSI lower layers framework, which is composed of ISO 13400-3 and ISO 13400-4[2].

The ISO 13400 series and ISO 14229-5[5] are based on the conventions specified in the OSI Service Conventions (ISO/IEC 10731)[1] as they apply for all layers and the diagnostic services.

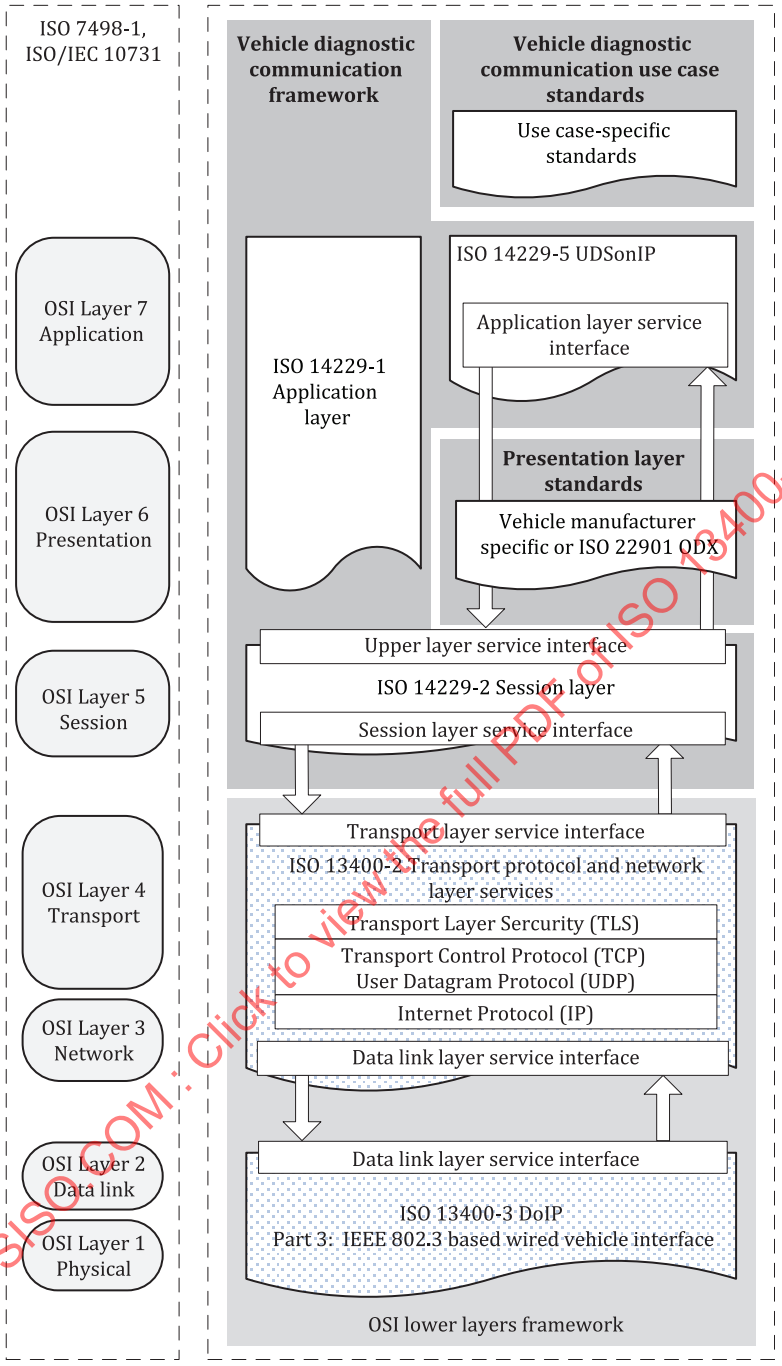


Figure 1 — DoIP document reference according to OSI model

Figure 2 illustrates vehicle network architecture schematics from a functional viewpoint.

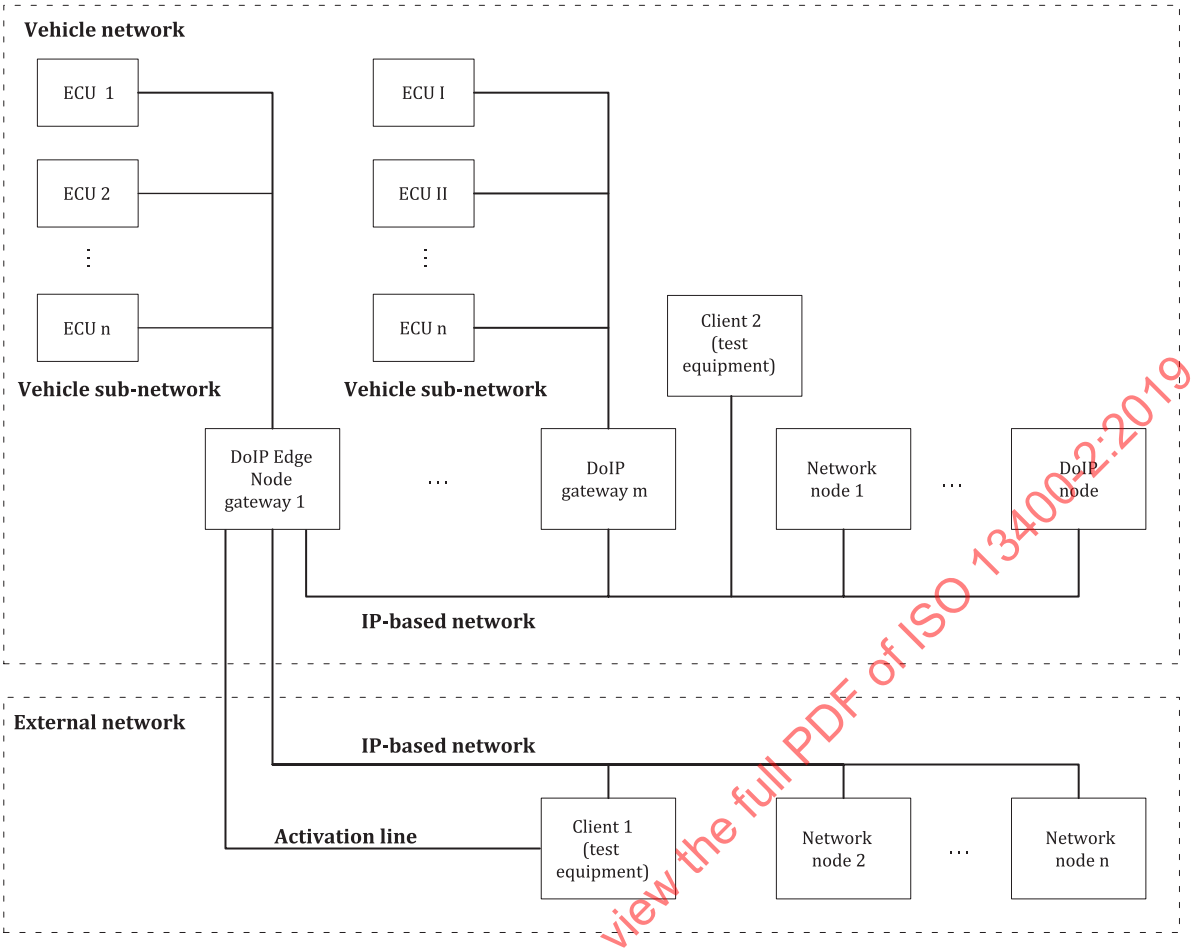


Figure 2 — Vehicle network architecture schematics (functional view)

This protocol standard is implemented by one or more DoIP entities, depending on the vehicle's network architecture. Figure 2 illustrates a client 1 (external client), which is connected to the DoIP edge node and a client 2 (internal client) in the vehicle's internal network. If not stated otherwise, the DoIP client entities are assumed to behave the same regardless to which network they are connected.

If necessary, this document distinguishes between an "internal client" and "external client" to apply a requirement or statement.

In this document, the requirements are assigned a unique number of the form "**X.DoIP-yyy**", allowing for easier requirement tracking and reference.

— X = OSI layer number; and

— DoIP-yyy = requirement number; and

— xL = x = OSI layer abbrevation[8 = APP, 7 = AL, 6 = PL, 5 = SL, 4 = TL, 3 = NL, 2 = DLL, 1 = PHY, 0 = SPP].

NOTE Requirements in this document are not numbered sequentially because the order of individual requirements changed during document development.

Requirements formulated as "The vehicle shall implement …" imply that this is a requirement for all DoIP entities to implement the required functionality if not explicitly stated otherwise. If multiple DoIP entities are present on a vehicle network, implementation details may differ slightly for each DoIP entity (e.g. for identification purposes), so that the client DoIP entity is able to identify the individual DoIP gateways that support this protocol standard.

Where reference is made to RFC documents, note that the forms “shall/shall not” are used to express requirements in these documents.

Road vehicles — Diagnostic communication over Internet Protocol (DoIP) —

Part 2:
Transport protocol and network layer services

1 Scope

This document specifies the requirements for secured and unsecured diagnostic communication between client DoIP entity and server(s) installed in the vehicle using Internet protocol (IP) as well as the transmission control protocol (TCP) and user datagram protocol (UDP). This includes the definition of vehicle gateway requirements (e.g. for integration into an existing computer network) and test equipment (client DoIP entity) requirements (e.g. to detect and establish communication with a vehicle).

This document specifies features that are used to detect a vehicle in a network and enable communication with the vehicle gateway as well as with its sub-components during the various vehicle states. These features are separated into two types: mandatory and optional.

This document specifies the following mandatory features:

— vehicle network integration (IP address assignment);

— vehicle announcement and vehicle discovery;

— vehicle basic status information retrieval (e.g. diagnostic power mode);

— connection establishment (e.g. concurrent communication attempts), connection maintenance and vehicle gateway control;

— data routing to and from the vehicle's sub-components;

— error handling (e.g. physical network disconnect).

This document specifies the following optional features:

— DoIP entity status monitoring;

— transport layer security (TLS);

— DoIP entity firewall capabilities.

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.

ISO/IEC 7498-1:1994, *Information processing systems — Open systems interconnection — Basic reference model*

ISO 13400-3, *Road vehicles — Diagnostic communication over Internet Protocol (DoIP) — Part 3: Wired vehicle interface based on IEEE 802.3*

ISO/IEC/IEEE 8802-3, *Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements — Part 3: Standard for Ethernet*

IETF RFC 768, *User Datagram Protocol*

IETF RFC 791:1981, *Internet Protocol — DARPA Internet Program — Protocol Specification*

IETF RFC 792, *Internet Control Message Protocol — DARPA Internet Program — Protocol Specification*

IETF RFC 793, *Transmission Control Protocol — DARPA Internet Program — Protocol Specification*

IETF RFC 826, *An Ethernet Address Resolution Protocol*

IETF RFC 1122, *Requirements for Internet Hosts — Communication Layers*

IETF RFC 2131, *Dynamic Host Configuration Protocol*

IETF RFC 2132, *DHCP Options and BOOTP Vendor Extensions*

IETF RFC 2460, *Internet Protocol, Version 6 (IPv6) — Specification*

IETF RFC 2375, *IPv6 Multicast Address Assignments*

IETF RFC 3315, *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*

IETF RFC 3484, *Default Address Selection for Internet Protocol version 6 (IPv6)*

IETF RFC 3927, *Dynamic Configuration of IPv4 Link-Local Addresses*

IETF RFC 4291, *IP Version 6 Addressing Architecture*

IETF RFC 4443, *Internet Control Message Protocol (ICMP v6) for the Internet Protocol Version 6 (IPv6) Specification*

IETF RFC 4492, *Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)*

IETF RFC 4702, *The Dynamic Host Configuration Protocol (DHCP) Client Fully Qualified Domain Name (FQDN) Option*

IETF RFC 4861, *Neighbor Discovery for IP version 6 (IPv6)*

IETF RFC 4862, *IPv6 Stateless Address Autoconfiguration*

IETF RFC 5246, *The Transport Layer Security (TLS) Protocol Version 1.2*

IETF RFC 8446:2018, *The Transport Layer Security (TLS) Protocol Version 1.3*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 7498-1 and the following apply.

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

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1
diagnostic power mode
abstract vehicle internal power supply state, which affects the diagnostic capabilities of all servers on the in-vehicle networks and which identifies the state of all servers of all gateway sub-networks that allow diagnostic communication

Note 1 to entry: The intent is to provide information to the client DoIP entity about whether diagnostics can be performed on the connected vehicle or whether the vehicle needs to be put into a different diagnostic power mode (i.e. technician interaction required). In this document, the following states are relevant: Not Ready (not all servers accessible via DoIP can communicate), Ready (all servers accessible via DoIP can communicate) and Not Supported (the Diagnostic Information Power Mode Information Request message is not supported).

3.2
DoIP edge node
host (3.4) inside the vehicle, where an Ethernet activation line in accordance with ISO 13400-3 is terminated and where the link from the first node/host in the external network is terminated

[SOURCE: ISO 13400-3:2011, 3.1.2, modified — definition editorially revised.]

3.3
DoIP entity certificate
certificate issued by an *intermediate CA* (3.5)to the DoIP entity presented during the TLS handshake to the client DoIP entity to verify the authenticity of this DoIP entity

3.4
host
node connected to the IP-based network

3.5
intermediate certificate authority
intermediate CA
authority, which issues subordinal certificates to another intermediate CA or DoIP entities

3.6
intermediate certificate
certificate either stored in the client DoIP entity or is presented during authentication together with the end node certificate to complete the chain of trust

3.7
invalid source address
address outside the reserved range for client(s) DoIP entity

3.8
logical address
address identifying a diagnostic application layer entity

3.9
network node
device connected to the IP-based network (e.g. Ethernet) and which communicates using Internet protocol but does not implement the DoIP protocol

Note 1 to entry: Some network nodes might also be connected to a *vehicle sub-network* (3.14), but they are not DoIP gateways as they don't implement the DoIP protocol. Consequently, these network nodes do not interact with (e.g. respond to) DoIP-compliant client DoIP entity.

3.10
root certificate authority
authority, which acts as the root of trust

Note 1 to entry: Typically issues *intermediate certificates* (3.6) to allow an *intermediate CA* (3.5) to further submit certificates.

3.11
root certificate
certificate created by the *root certificate authority* (3.10) and used as the trust anchor

Note 1 to entry: It is securely stored and used by all entities that wants to validate end node certificates (e.g. from the DoIP entity) together with all necessary *intermediate certificates* (3.6) in the chain of trust.

3.12
socket
unique identification, as defined in IETF RFC 147, to or from which information is transmitted in the network

3.13
unknown source address
address not listed in the connection table entry

3.14
vehicle sub-network
network not directly connected to the IP-based network

Note 1 to entry: Data can only be sent to and from a vehicle sub-network through the connecting DoIP gateway.

4 Symbols and abbreviated terms

4.1 Symbols

<d>	payload length, given in bytes
<m>	number of concurrent DoIP TCP sessions that the client DoIP entity is required to support in order to connect to one or more DoIP entities
<n>	number of concurrent DoIP TCP sessions that the DoIP entity needs to support in order to accept 1 to N concurrent connections to one or more items of the client DoIP entity
<u>, <v>	number of individual servers in a vehicle sub-network
<w>	number of individual DoIP gateways in a vehicle network
<x>	number of individual in-vehicle network nodes
<y>	number of individual vehicle DoIP nodes in a vehicle network
<z>	number of individual vehicle external network nodes

4.2 Abbreviated terms

AL	application layer
Alt	alternative
APP	application
ARP	address resolution protocol
ASCII	American standard code for information interchange
Auto-MDI(X)	automatic medium-dependent interface crossover
CA	certificate authority

CAN	controller area network
CF	consecutive frame
DHCP	dynamic host control protocol
DLL	data link layer
DNS	domain name system
DoIP	diagnostic communication over Internet Protocol
EID	entity identification
FF	first frame
FMI	failure mode indicator
GID	group identification
GUI	graphical user interface
GW	gateway
IANA	Internet assigned numbers authority
ICMP	Internet control message protocol
IETF RFC	Internet Engineering Task Force Request for Comments
IP	Internet protocol
IPv4	Internet protocol version 4 (see IETF RFC 791)
IPv6	Internet protocol version 6 (see IETF RFC 2460)
MAC	media access control
MSC	message sequence chart
MTU	maximum transport unit
NDP	neighbour discovery protocol
NL	network layer
OSI	open systems interconnection
PKI	public key Infrastructure
SA	source address
SDU	service data unit
SF	single frame
SPN	suspect parameter number
SPP	service primitive parameter
TA	target address

TCP	transmission control protocol
TL	transport layer
TLS	transport layer security
UDP	user datagram protocol
VIN	vehicle identification number (see ISO 3779)
VM	vehicle manufacturer
XOR	exclusive or

5 Conformance

This document is based on the conventions discussed in the OSI Service Conventions as specified in ISO/IEC 10731[1] as they apply to diagnostic services.

6 DoIP introduction

6.1 General information

This subclause gives an example of a standard workflow of a straightforward DoIP session. In order to keep this introduction as helpful as possible for a reader new to DoIP, exceptions and errors that might occur during a DoIP session are not covered here. Two possible network environments—networked and directly connected—are explained. The figures provide a better understanding of the comprised DoIP components, mechanisms and sequences that allow a proper DoIP session.

As only the connection and the vehicle discovery (see 7.4) differ between the direct connection and the networked scenarios, the homogeneous parts of the DoIP session are described in Figure 11 for both scenarios.

6.2 Connection establishment and vehicle discovery

6.2.1 Direct connection scenario

In a direct connection scenario with no networking infrastructure, a “crossover” Ethernet cable is used or Auto-MDI(X) is supported by the Ethernet controller by either the client DoIP entity or the DoIP entity (server), in order to directly connect the vehicle to the client DoIP entity.

It is assumed that, in such a scenario, no DHCP server is present. Thus, although initiated, the DHCP process is not successful. Rather, a locally valid IP address is determined by the auto-configuration mechanism and afterwards configured for both interfaces involved.

As soon as the DoIP entity’s interface is configured with the obtained IP address, the DoIP entity broadcasts its vehicle identification number (VIN), entity identification (EID), group identification (GID), and logical address through a vehicle announcement message (see 7.4). The message is broadcasted (UDP) three times with the destination port UDP_DISCOVERY.

Depending on whether the client DoIP entity is configured in time for TCP/IP communication to receive the initial vehicle announcement messages, the client DoIP entity may poll for a vehicle using the vehicle identification request message. The Auto-IP mechanism might be delayed on the client DoIP entity as some operating systems start the Auto-IP only after DHCP has failed. As the DoIP entity initiates both mechanisms in parallel, it is likely that its IP configuration is completed quickly and the client DoIP entity does not receive the initial vehicle announcement.

Figure 3 shows the connection and vehicle discovery in a direct connection scenario.

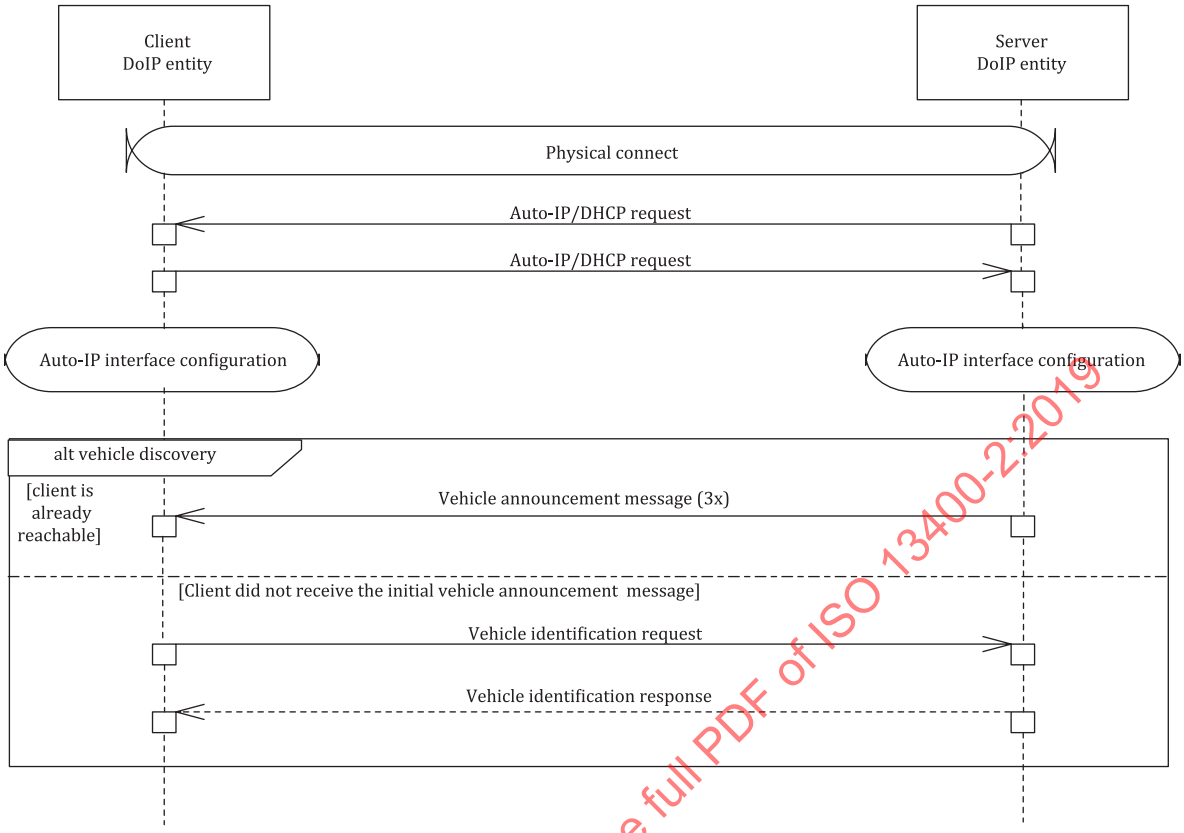


Figure 3 — Connection and vehicle discovery in a direct connection scenario

6.2.2 Network connection scenario

The connection and vehicle discovery process is slightly different in a networked scenario. The physical connections to the network are not necessarily synchronized in time. Accordingly, the points in time when the interfaces are configured and accessible for a TCP/IP connection attempt might differ significantly.

In certain network scenarios, there can be multiple vehicles sending vehicle announcement messages. If the vehicle's DoIP entity does not send a vehicle announcement message, the client DoIP entity can poll for the vehicle announcement message by sending vehicle identification request messages.

Figure 4 depicts the connection and vehicle discovery in a networked scenario.

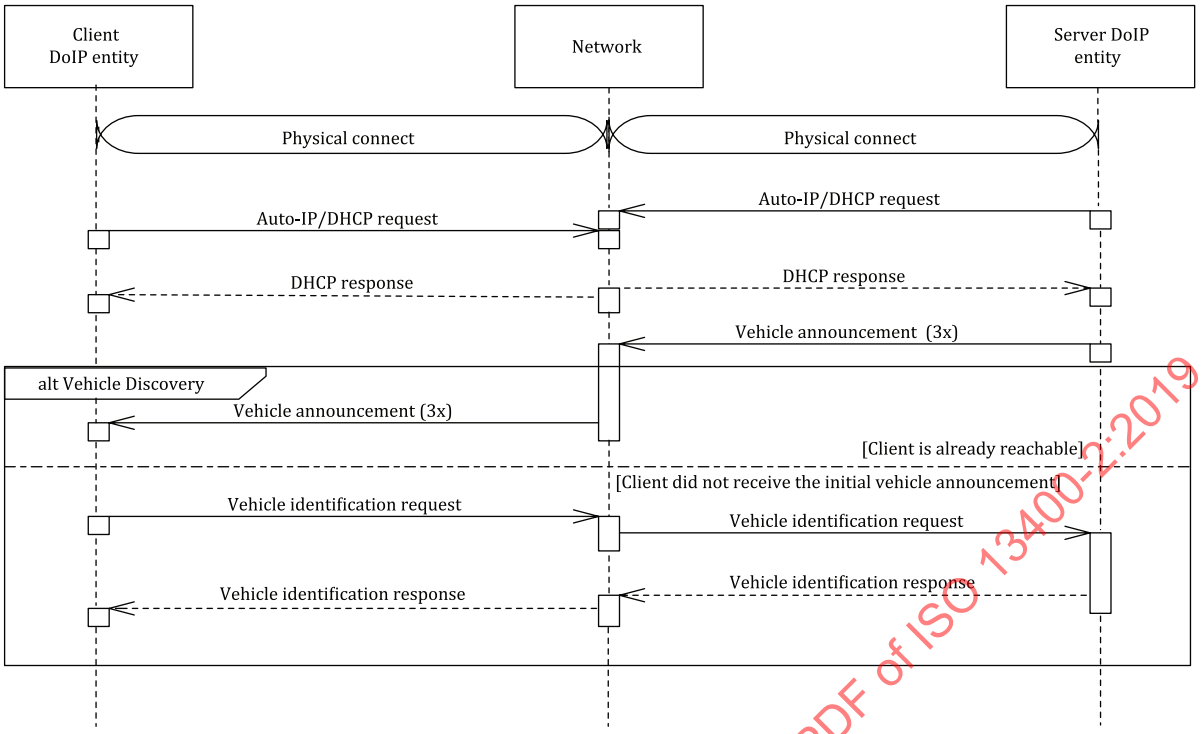


Figure 4 — Connection and vehicle discovery in a networked scenario

6.2.3 Internal tester scenario (optional)

Use cases like "Over-The-Air" OTA software update or remote diagnostics use vehicle internal test equipment. Internal test equipment can omit dynamic IP address assignment via DHCP or AutoIP to enable minimal interface startup times by using static IP address configuration. This also applies to vehicle announcements on vehicle-internal IP interfaces .

6.2.4 Unsecured DoIP session

The step “add vehicle to list” (see Figure 5) is not covered by this document and thus it is not mandatory or may not even be necessary. Nevertheless, it is likely, that the vehicle announcement message broadcast in the previous step is processed in some way. For example, the vehicle somehow indicates to be “ready” or an automated process is initiated based on the information that a vehicle is now available for a DoIP session.

Although in the networked scenario there is still the networking equipment between the client and the DoIP entity, the communication is now logically directly between the two communication endpoints. Thus, no network is shown in Figure 5.

The first step in order to initiate a connection between the client DoIP entity and the DoIP entity in the vehicle is to open a socket (destination port is TCP_DATA). This is done prior to any message exchange. Therefore, a DoIP entity provides the resources to handle the incoming communication request (e.g. socket resources). The DoIP entity provides sufficient resources to handle the specified number of concurrently supported DoIP sessions (<n>) plus one extra socket (see DoIP-002). If more than <n + 1> connection attempts arrive at the same time, it is possible that no more resources are free and the <n + 2nd> connection attempt is refused (because there are no longer any sockets in the listening state because of DoIP protocol handling).

Once a socket is established, some initializing steps are performed. An initial inactivity timer (see 12.6.3) and a general inactivity timer (see 12.6.2) is assigned and started. Additionally, it is necessary to ensure that no arriving data, except the routing activation request message, is routed or processed by setting the connection state to “initialize” (see 12.6.1.3). All subsequent messages are exchanged

through this TCP_DATA socket. This connection handling is also applied by the DoIP entity in case of secure TLS-based communication and the corresponding TCP_DATA TLS socket (see 6.2.5).

To activate routing on the initialized connection, the client sends a routing activation request message (see 12.5.2) to the DoIP entity. If the client DoIP entity is eligible and if there are fewer than <n> active connections registered, the corresponding initial timer is stopped and—assuming that no additional authentication or confirmation or secure TLS connection is required—the socket state changes to "registered [routing active]". Now valid DoIP messages (e.g. DoIP diagnostic messages) are routed or processed. This is reported to the client DoIP entity by a positive routing activation response message. The general inactivity timer is restarted and remains active.

When receiving any kind of data, the DoIP entity first calls the DoIP header handler. If the payload consists of a diagnostic message (identified through the payload type 8001_{16} in the generic DoIP header, see 9.5), the diagnostic message handler is called to process the payload.

When a diagnostic message arrives, the DoIP confirmation is sent to the calling client DoIP entity immediately after the message has successfully passed the diagnostic message handler (confirmation acknowledgement), in essence the message has passed through the corresponding internal routing mechanism but has not yet been processed by the DoIP gateway or forwarded to the final non DoIP server.

In the case of an ISO 14229-1 conform diagnostic message payload, the destination server DoIP entity sends a diagnostic response back to the client DoIP entity. This behaviour is described by the corresponding diagnostic protocol encapsulated by the DoIP message and thus is not within the scope of this document.

When a connection is no longer required by the client DoIP entity, it always closes through TCP/IP protocol mechanisms. The DoIP entity then initiates a finalization process for the connection. That finalization frees the corresponding resources so that the socket is available for a new connection. If the connection is not closed, the resources are freed after a timeout based on the general inactivity timer or after the performance of an alive check.

Figure 5 depicts the unsecured DoIP session example.

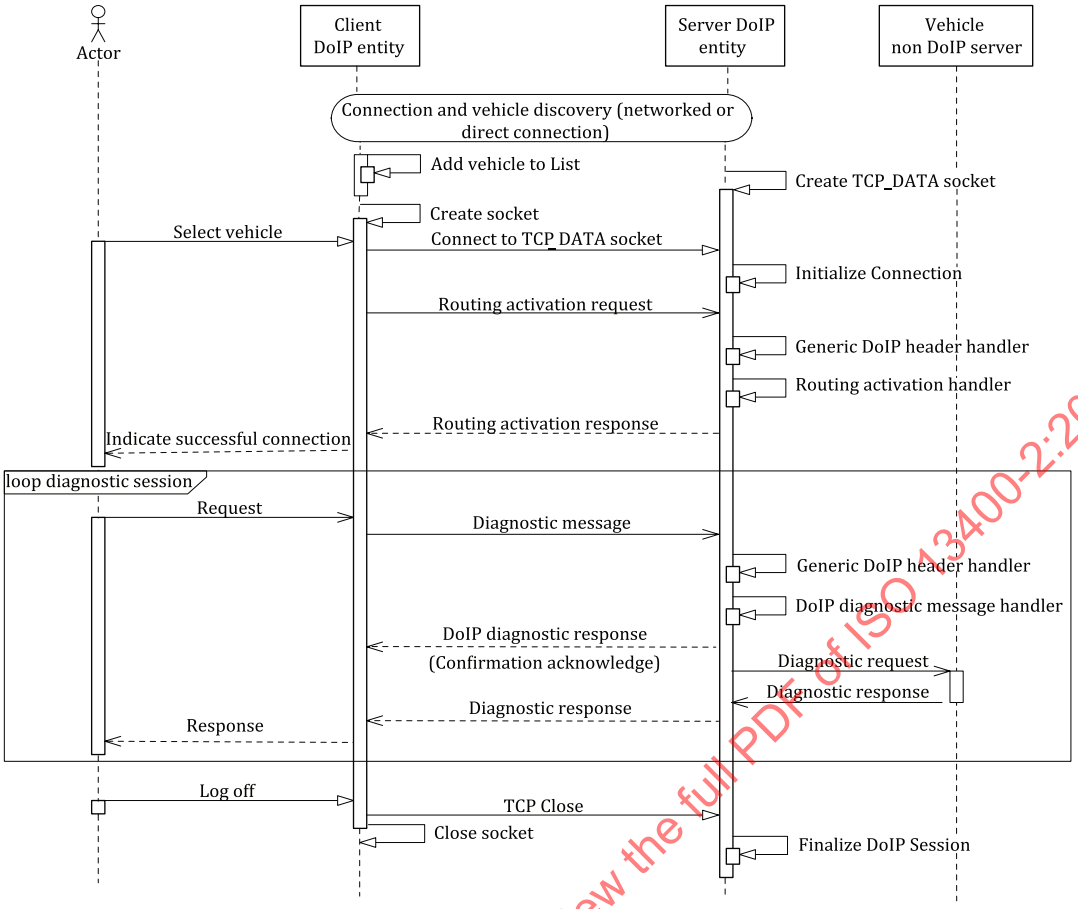


Figure 5 — Unsecured DoIP session example

6.2.5 Secured (TLS) DoIP session

For a secured TCP connection, the TLS dedicated TCP_DATA port is used. As for the unsecured DoIP session case, the first step in order to initiate a secure TLS connection between the client DoIP entity and the DoIP entity, is to open a TLS socket (destination port is TLS TCP_DATA). This is done prior to any message exchange. Therefore, a DoIP entity provides the resources to handle the incoming communication request (e.g. socket resources). The DoIP entity provides sufficient resources to handle the specified number of concurrently supported DoIP sessions secured with TLS (<k>) plus one extra socket (see [DoIP-159]). If more than <k + 1> connection attempts do arrive at the same time, it is possible that no more resources are available and the <k + 2nd> connection attempt is refused (because there are no longer any sockets in the listening state rather than because of DoIP protocol handling).

Once a socket is established, the TLS protocol specific handshake initializing steps is performed by the client DoIP entity and the DoIP entity. After the TLS handshake is successfully completed, all subsequent messages are exchanged through this TLS TCP_DATA socket (e.g. routing activation and DoIP diagnostic messages).

Figure 6 shows the DoIP session secured with TLS example.

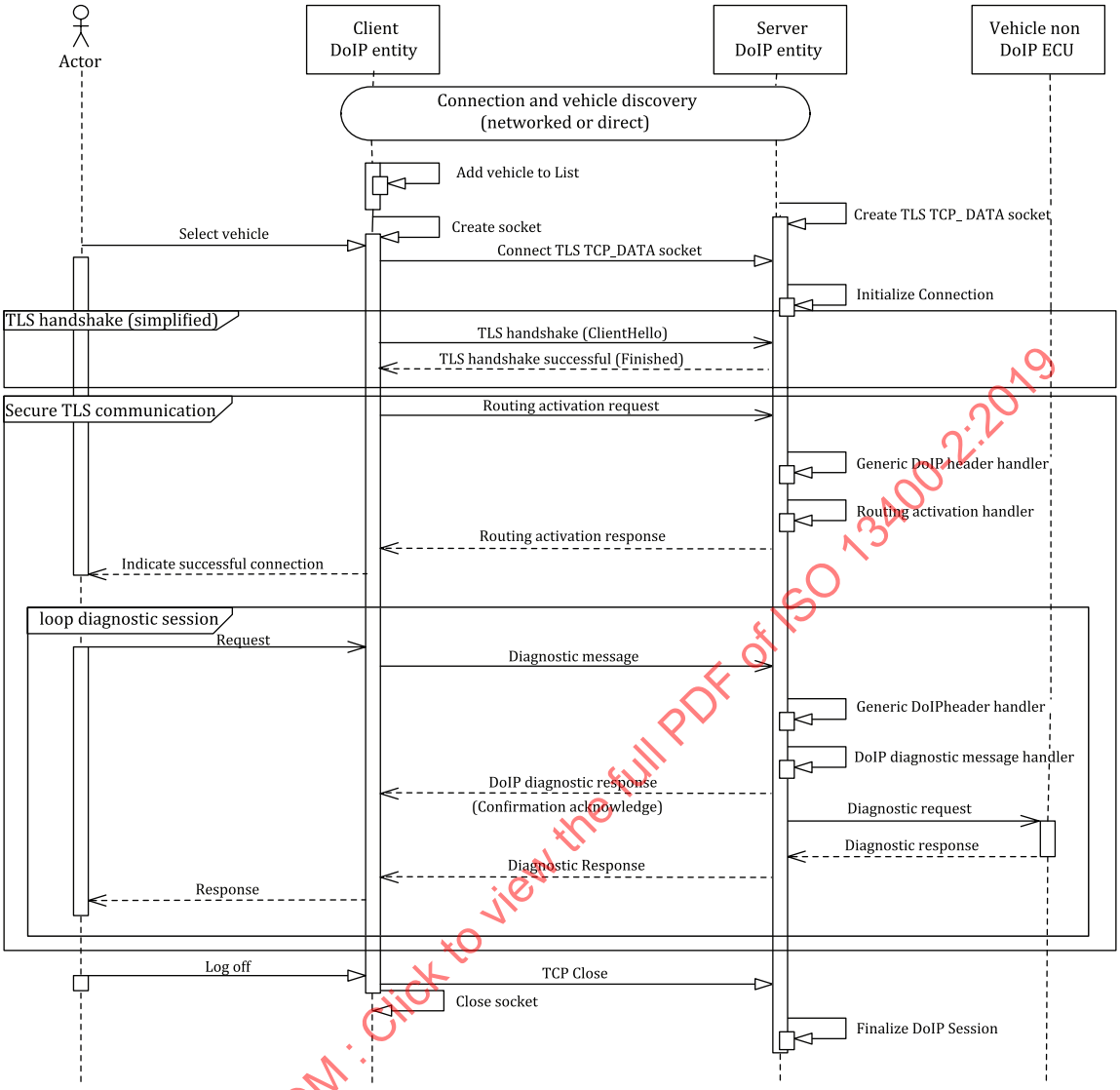


Figure 6 — Secured DoIP session with TLS example

When the secured TCP connection is no longer required by the client DoIP entity, it always closes cleanly through TLS and TCP/IP protocol mechanisms (see e.g. TLS 1.2 RFC 5246:2008, 7.2.1, TLS 1.3 RFC 8446:2018, 6.1). The DoIP entity also destroys any TLS related information of the current session and make available the corresponding resources so that the TLS socket is available for a new connection.

6.3 Vehicle network integration

6.3.1 Vehicle identification

Vehicle identification specifies how a vehicle and its DoIP entities can be discovered and associated with their IP addresses on the network.

A vehicle is usually identified by its VIN. In manufacturing or in after-sales environments, several DoIP entities may be installed on the same vehicle, but the vehicle-specific VIN is not yet configured at this point in time. In order to associate newly installed and un-configured DoIP entities with a vehicle, the group ID (GID) can be used instead of the VIN.

6.3.2 Multiple vehicles in a single network

This subclause gives an example of a sequence by which the external client DoIP entity may be able to identify and group server DoIP entities of all connected vehicles within a network.

Figure 7 shows an example of a simplified identification sequence performed by the client DoIP entity. When a vehicle is connected to the DoIP network and the IP address allocation is completed (see Figure 5), the DoIP entities send out vehicle announcements after waiting for A_DoIP_Announce_Wait.

If the client DoIP entity is connected to the DoIP network at a later time, it should trigger vehicle announcement/identification responses by sending a broadcast vehicle identification request.

The server DoIP entities in all vehicles respond to a vehicle identification request within A_DoIP_Ctrl.

If a vehicle announcement/vehicle identification is received by the client DoIP entity and contains a VIN/GID synchronization status incomplete message (10_{16}), meaning that the VIN or GID is *not* synchronized with all server DoIP entities in the vehicle, the client DoIP entity starts a vehicle discovery timer for this vehicle (identified by the VIN/GID given by the VIN/GID master in its vehicle announcement/vehicle identification response).

This mechanism allows the VIN/GID master to notify the client DoIP entity when some entities need more time for VIN/GID synchronization. When the vehicle discovery timer expires, another vehicle identification request is sent to all those DoIP entities, which reported VIN/GID invalid in their initial vehicle announcement/identification responses.

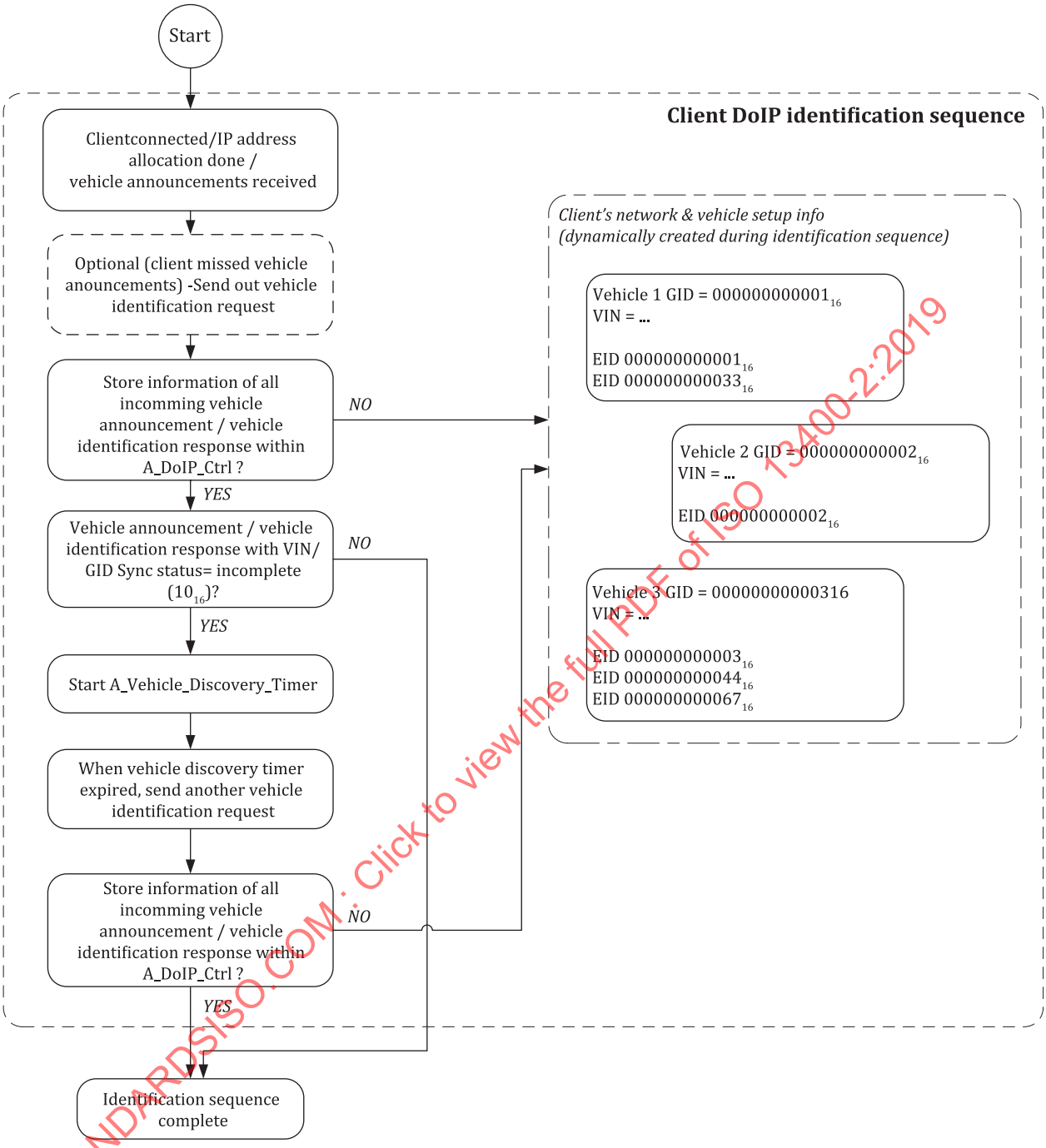


Figure 7 — Example of a simplified identification sequence of client

6.4 Communication examples using message sequence charts

Several message sequence charts (MSCs) show the most common communication scenarios of client DoIP entity communication with server DoIP entities of a vehicle.

6.5 IP-based vehicle communication protocol — General information

The tables in 7.4 to 7.6, 9.2 to 9.6, and 12.5 describe each message under the following column headings:

— Item:

This is a short name for the message element. This name is used when the message element is referenced in this document.

— Pos.:

This is the position (byte number) of each individual message element in a DoIP message. The byte position always starts with zero (0) and is counted from the beginning of the PDU.

— Len:

This is the length (number of bytes) of the respective message element.

— Description:

This column contains a more detailed description of each individual message element and its purpose.

— Values:

This column lists the supported value range and meaning of individual values of the respective message element.

— Support:

This column contains information on whether a specific message or message element is supported by a DoIP entity. Even if a message itself is defined as optional, it may contain mandatory elements to be implemented if the message itself is supported.

— Port and protocol:

This column specifies on which underlying protocol a specific payload type is supported and which port it uses.

7 Application (APP) requirements

7.1 APP implementation of DoIP requirements

REQ	**8.DoIP-108 APP –** This document's **implementation**
Each DoIP entity on a vehicle network shall implement the protocol requirements as specified in this document.	

7.2 APP data transmission order

REQ	**8.DoIP-147 APP – Big-endian network byte order**
The big-endian network byte order of IP shall be used for DoIP messages in accordance with IETF RFC 791:1981 , Annex B.	

7.3 APP DoIP entity synchronization of a vehicle's GID

A group identification (GID) is a decentralized approach for identifying multiple DoIP entities within one vehicle. This implies that there is a VIN/GID master (e.g. the DoIP edge node) from which all other DoIP entities receive the VIN/GID during a synchronization process. As this synchronization process

usually requires some time (e.g. after a new DoIP entity is added to the vehicle) invalidity values are defined (see Table 1) for use by the DoIP entities until the VIN/GID synchronization is finished.

A detailed specification of VIN/GID synchronization between DoIP entities is outside the scope of this document and is left to the VM's discretion.

REQ	8.DoIP-143 APP – DoIP entity synchronization of a vehicle's GID
Each DoIP entity shall support the synchronization of a vehicle's GID if more than one DoIP entity is present within the vehicle and if the availability of a valid VIN cannot always be guaranteed to be configured for every DoIP entity.	

NOTE One possible way to ensure a globally unique GID is to use the GID masters MAC address.

Figure 8 describes schematically the VIN/GID synchronization and identification of two separate DoIP entities within the same vehicle.

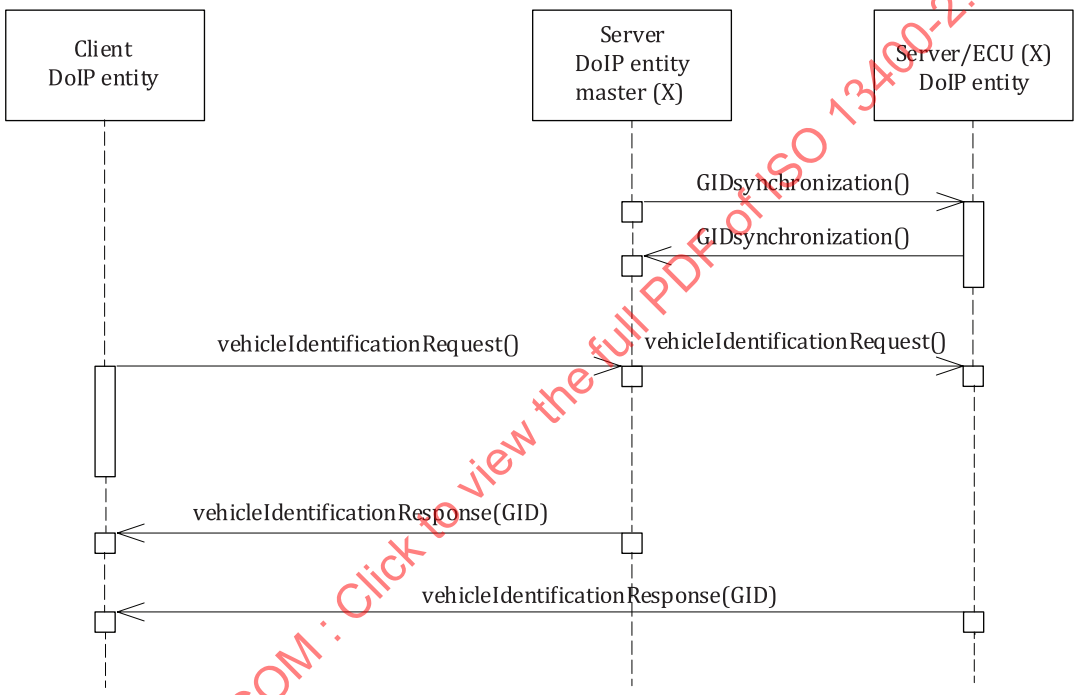


Figure 8 — Example of vehicle identification with VIN/GID synchronization

Table 1 defines the values if parameter value is not set.

Table 1 — Vehicle identification parameter values (value not set)

Parameter	Length	Values
VIN	17	00_{16} or FF_{16}
Logical address	2	0000_{16} or $FFFF_{16}$
Entity ID (EID)	6	00_{16} or FF_{16}
Group ID (GID)	6	00_{16} or FF_{16}

Figure 9 shows the sequence for connecting client DoIP entity to the vehicle and the IP address allocation process.

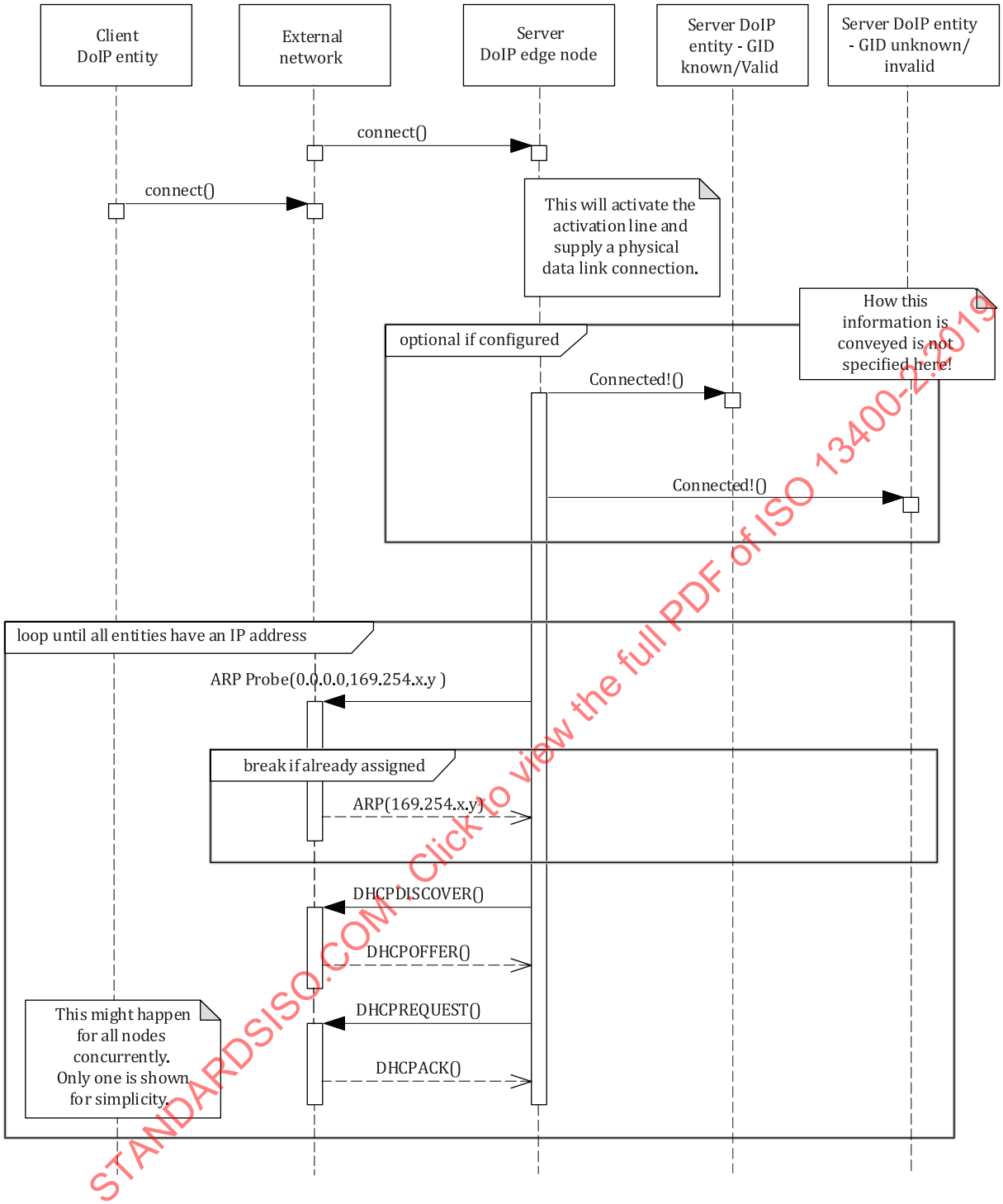
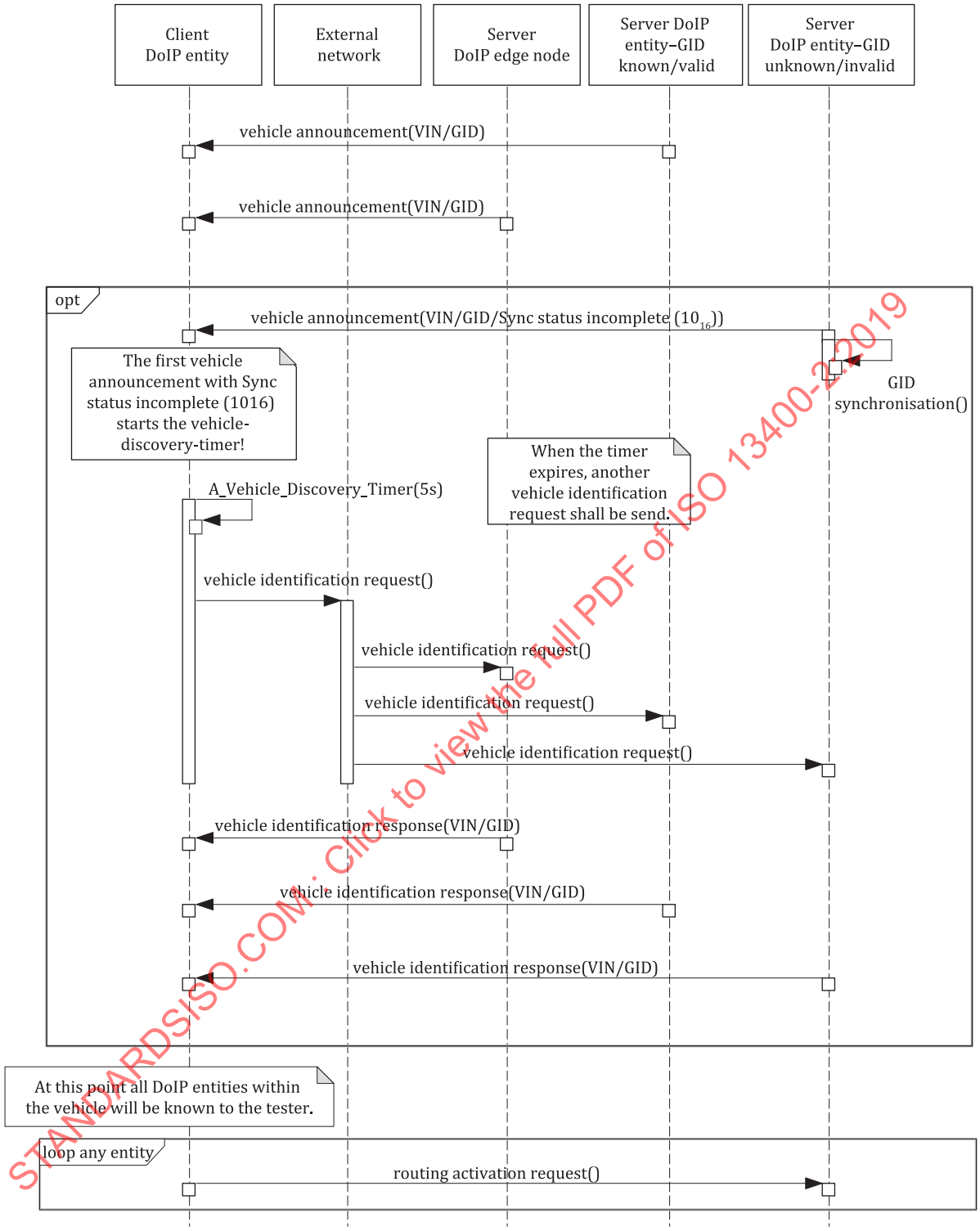


Figure 9 — Detailed vehicle identification without VIN, GID synchronisation

Figure 10 describes in more detail the complete decentralized approach to identifying multiple DoIP entities.



NOTE The scenario in Figure 10 does not cover the case in which vehicles are connected to the DoIP network once the vehicle discovery timer has already started.

Figure 10 — Detailed vehicle identification with VIN/GID synchronisation

7.4 APP vehicle identification and announcement request message

The vehicle identification request and vehicle announcement messages specify the requirements to identify a vehicle or its DoIP entities in a network. In order for the client DoIP entity to communicate with a DoIP entity, it needs to know its IP address and in which vehicle it is installed. The client DoIP

entity then knows the IP addresses, the vehicle identification request message is used to retrieve the VIN/GID and DoIP entities logical addresses from a vehicle (see 6.3.1). Therefore, the following scenarios are supported:

— vehicle with VIN not yet configured (e.g. during assembly phase or after reprogramming);

— vehicle with VIN configured and VIN/EID/GID unknown to the client DoIP entity;

— vehicle with VIN configured and VIN/EID/GID known to the client DoIP entity;

— multiple DoIP entities installed on the same vehicle;

— IP addresses of DoIP entities known.

In the case that IP addresses are unknown and VINs are not yet configured, it is impossible to associate DoIP entities with a single vehicle based on the VIN. An alternative approach to this association problem is described in 6.3.1.

Figure 11 depicts the common vehicle announcement and identification sequence of DoIP entities (gateway/node DoIP entity and client DoIP entity) announcing their presence or being identified using the specified requests and responses.

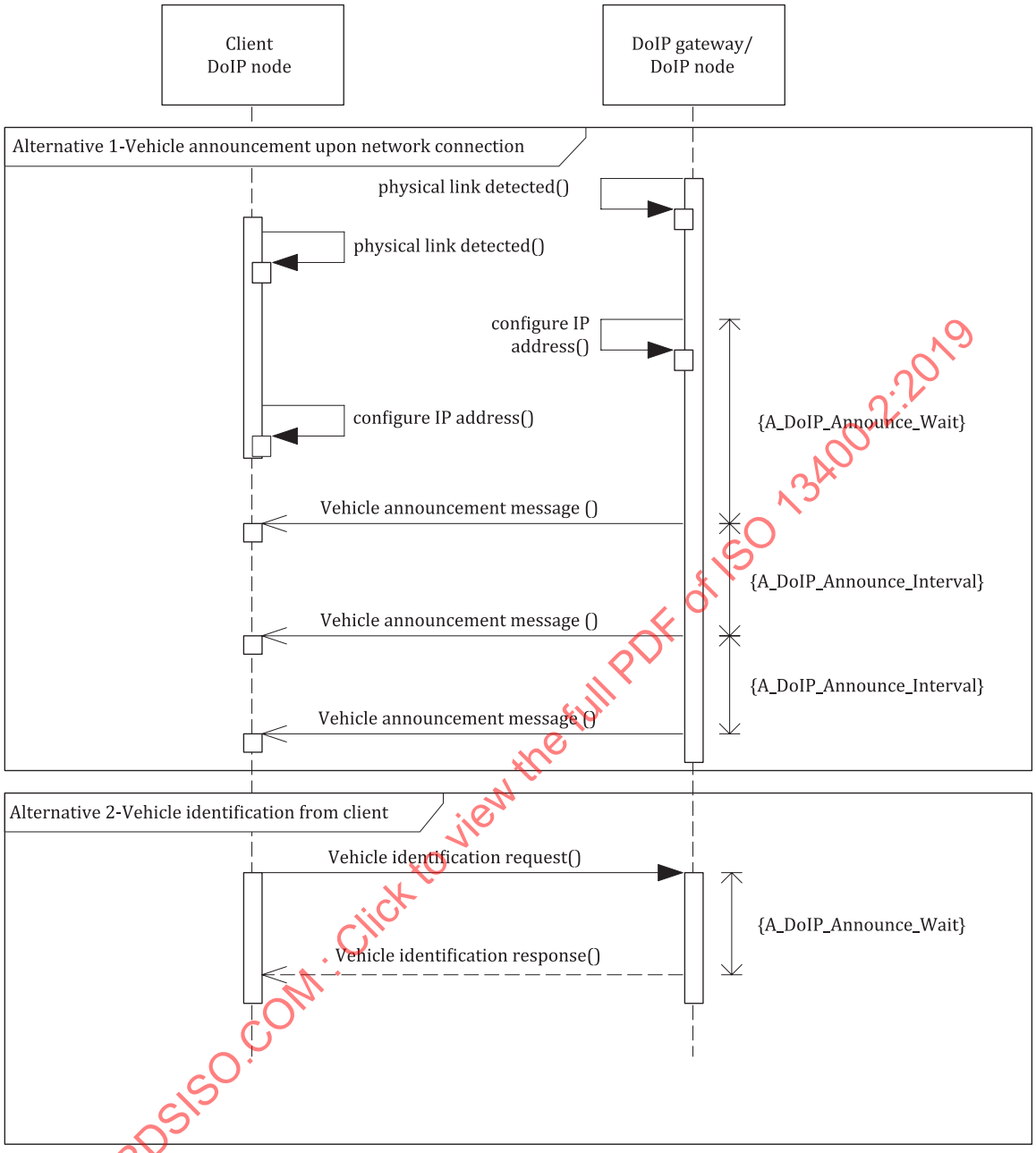


Figure 11 — Vehicle announcement and identification sequence

Figure 12 shows a sequence followed by the TCP_DATA socket handler, which is handling two concurrent sockets when a third connection is requested with a new routing activation request.

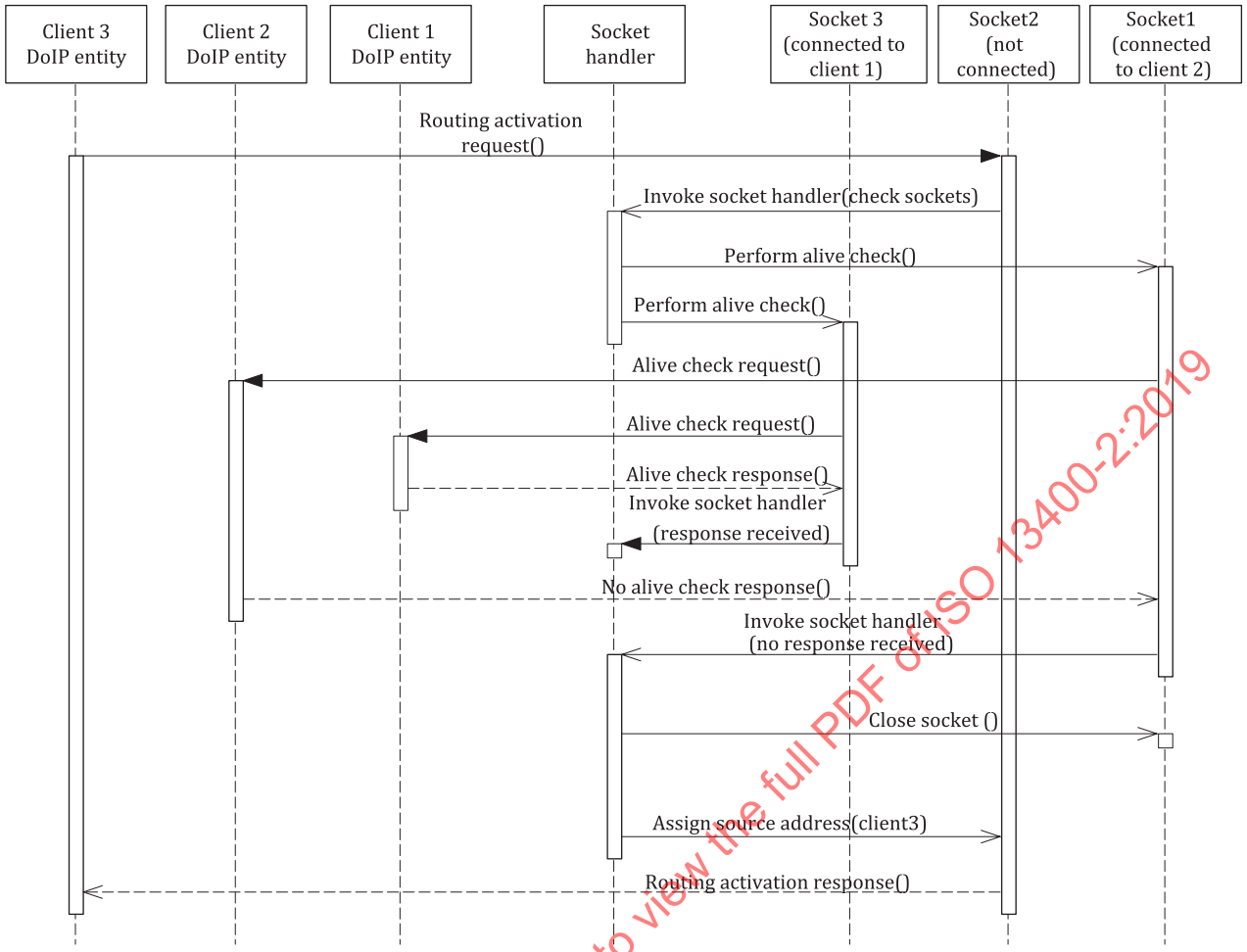


Figure 12 — Socket handler with two concurrent sockets and a third connection attempt

REQ	8.DoIP-046 APP – DoIP entity vehicle identification request message
Each DoIP entity shall support the vehicle identification request message as specified in Table 2.	

Table 2 — Payload type vehicle identification request message — No message parameters

Item	Pos.	Len.	Description	Values	Support condition
No message parameters					

REQ	8.DoIP-047 APP – DoIP entity vehicle identification request message with EID
If supported each DoIP entity shall implement the vehicle identification request message with additional EID parameter as specified in Table 3.	

Table 3 — Payload type vehicle identification request message with EID

Item	Pos.	Len.	Description	Values	Support condition
EID	0	6	This is the DoIP entity's unique ID (e.g. network interface's MAC address) that shall respond to the vehicle identification request message.	If MAC address is used, it shall be in accordance with IEEE EUI-48.	mandatory

REQ	8.DoIP-048 APP – DoIP entity vehicle identification request message with VIN
Each DoIP entity shall support the vehicle identification request message with an additional VIN parameter as specified in Table 4.	

Table 4 — Payload type vehicle identification request message with VIN

Item	Pos.	Len.	Description	Values	Support condition
VIN	0	17	This is the vehicle's identification number as specified in ISO 3779. This parameter is only present if the client DoIP entity intends to identify the DoIP entities of an individual vehicle, the VIN of which is known to the client DoIP entity.	ASCII	mandatory

REQ	8.DoIP-049 APP – DoIP entity vehicle announcement/identification response
Each DoIP entity shall support the vehicle announcement/identification response message as specified in Table 5.	

Table 5 — Payload type vehicle announcement/identification response message

Item	Pos.	Len.	Description	Values	Support condition
VIN	0	17	This is the vehicle's VIN as specified in ISO 3779. If the VIN is not configured at the time of transmission of this message, this should be indicated using the invalidity value specified in Table 1. In this case, the GID is used to associate DoIP nodes with a certain vehicle (see 6.3.1).	ASCII See Table 1.	mandatory
Logical Address	17	2	This is the logical address that is assigned to the responding DoIP entity (see 7.8 for further details). The logical address can be used, for example, to address diagnostic requests directly to the DoIP entity.	See Table 13.	mandatory
EID	19	6	This is a unique identification of the DoIP entities in order to separate their responses even before the VIN is programmed to or recognized by the DoIP devices (e.g. during the vehicle assembly process). It is recommended that the MAC address information of the DoIP entity's network interface be used (one of the interfaces if multiple network interfaces are implemented).	If MAC address is used, it shall be in accordance with IEEE EUI-48.	mandatory
GID	25	6	This is a unique identification of a group of DoIP entities within the same vehicle in the case that a VIN is not configured for that vehicle. The VIN/GID synchronization process between DoIP nodes of a vehicle is defined in 6.3.1. If the GID is not available at the time of transmission of this message, this shall be indicated using the specific invalidity value as specified in Table 1.	See Table 1.	mandatory

Table 5 *(continued)*

Item	Pos.	Len.	Description	Values	Support condition
Further action required	31	1	This is the additional information to notify the client DoIP entity that there are either DoIP entities with no initial connectivity or that a centralized security approach is used.	See Table 6.	mandatory
VIN/GID sync. status	32	1	This is the additional information to notify the client DoIP entity that all DoIP entities have synchronized their information about the VIN or GID of the vehicle.	See Table 7.	optional

NOTE 1 The information that indicates whether further actions are required can be used to signal that certain in-vehicle synchronization procedures have not yet finished and/or additional steps are required (e.g. security measures) in order to allow all DoIP nodes to announce their presence on the network.

REQ	8.DoIP-050 APP – DoIP entity vehicle announcement message
Each DoIP entity shall send the vehicle announcement message as specified in Table 5 A_DoIP_Announce_Num times with A_DoIP_Announce_Interval seconds inter-message time between each transmission starting immediately after configuration of a valid IP address.	

NOTE 2 The reason for transmitting this message multiple times is to compensate for the fact that there is no guarantee that the message is delivered correctly over the network, due to the use of UDP. Multiple transmissions increase the probability that at least one message is received correctly by the client DoIP entity.

Table 6 defines the further action code values.

Table 6 — Definition of further action code values

Value	Description	Support
00_{16}	no further action required	mandatory
01_{16} to $0F_{16}$	reserved by this document	mandatory
10_{16}	routing activation required to initiate central security	optional
11_{16} to FF_{16}	available for additional VM-specific use	optional

REQ	8.DoIP-144 APP – DoIP entity vehicle announcement/identification response message action code of 10_{16}
When a vehicle announcement/identification response message with a further action code of 10_{16} (see Table 6) is received from a DoIP entity, the client DoIP entity shall send a routing activation request message (see Table 46) with the activation type set to $E0_{16}$ (see Table 47) to that DoIP entity, which determines the specific action from the VM-specific field in the routing activation response message (see Table 48).	

Table 7 defines the VIN/GID synchronization status code values.

Table 7 — Definition of VIN/GID synchronization status code values

Value	Description	Support
00_{16}	VIN and/or GID are synchronized	mandatory
01_{16} to $0F_{16}$	reserved by this document	mandatory
10_{16}	incomplete: VIN and GID are not synchronized	mandatory
11_{16} to FF_{16}	reserved by this document	mandatory

REQ	8.DoIP-125 APP – DoIP entity vehicle announcement UDP target IPv4 address
In the case of a vehicle announcement (not the vehicle identification response), the UDP message shall always be sent with the target IPv4 address set to the limited broadcast address.	

REQ	8.DoIP-155 APP – DoIP entity vehicle announcement UDP target IPv6 address
In the case of a vehicle announcement (not the vehicle identification response), the UDP message shall always be sent with the target IPv6 address set to the link-local scope multicast address ($FF02_{16}$::1) as described in IETF RFC 2375.	

REQ	8.DoIP-123 APP – DoIP entity identifiable by VIN, the EID or both
Each DoIP entity shall be uniquely identifiable by either the VIN, the EID or both at any time.	

REQ	8.DoIP-142 APP – DoIP entity identifiable at least by EID and GID
If it cannot be guaranteed that a vehicle can be identified by the VIN at any time, support for EID and GID shall be provided.	

Figure 13 shows the generation of vehicle identification response messages, depending on the payload type of the vehicle identification request.

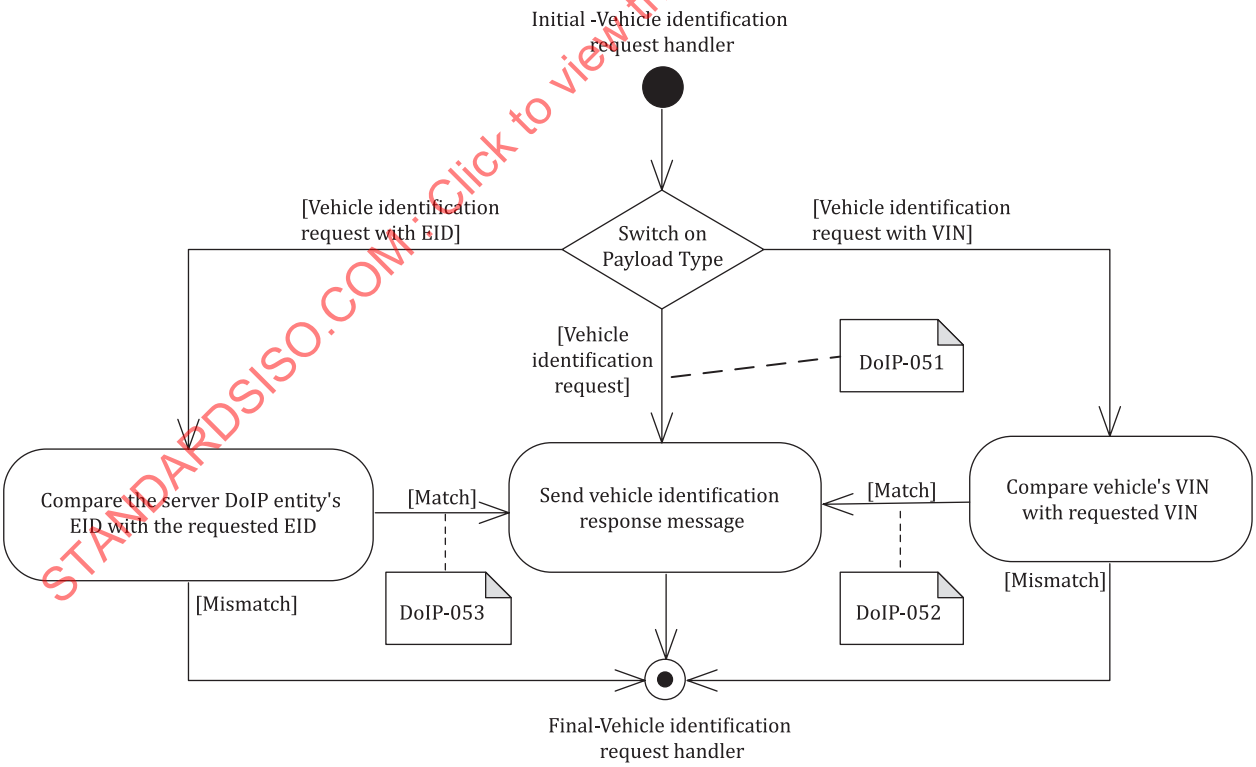


Figure 13 — Vehicle identification request handler

REQ	8.DoIP-051 APP – DoIP entity A_DoIP_Announce_Wait
Each DoIP entity shall send a delayed (A_DoIP_Announce_Wait as specified in Table 12) vehicle identification response message as specified in Table 5 after receipt of a vehicle identification request message as specified in Table 2.	

NOTE 3 The additional delay before responding to a vehicle identification request is necessary in order to avoid UDP packet bursts on the network if many DoIP entities are connected to the same network. In such a case, the random delay of the vehicle identification announcement response allows for UDP packets that were dropped due to high network utilization to reach the client DoIP entity on subsequent vehicle identification request broadcasts.

REQ	8.DoIP-052 APP – DoIP entity vehicle identification response after receipt of VIN
Each DoIP entity shall send the vehicle identification response message as specified in Table 5 after receipt of a vehicle identification request message with VIN (see Table 4), if the VIN from the request message matches the DoIP entity's programmed VIN.	

REQ	8.DoIP-053 APP – DoIP entity vehicle identification response after receipt of EID
Each DoIP entity shall send the vehicle identification response message as specified in Table 5 after receipt of a vehicle identification request message with EID (see Table 3), if the EID from the request message matches the DoIP entity's EID (e.g. one of the MAC addresses, if the DoIP entity implements multiple network interfaces).	

7.5 APP diagnostic power mode information request and response

This payload type serves the purpose of retrieving the diagnostic power mode of a vehicle. This information may be used by client DoIP entity, for example to verify whether the vehicle is in diagnostic power mode, which allows reliable diagnostics to be performed on the vehicle's components.

REQ	8.DoIP-116 APP – DoIP entity diagnostic power mode information request
Each DoIP entity shall support the diagnostic power mode information request as specified in Table 8.	

Table 8 — Diagnostic power mode information request

Item	Pos.	Len.	Description	Values	Support
No additional message element					

REQ	8.DoIP-117 APP – DoIP entity diagnostic power mode information response
Each DoIP entity shall support the diagnostic power mode information response as specified in Table 9.	

REQ	8.DoIP-118 APP – DoIP entity diagnostic power mode information response within A_DoIP_Ctrl
A DoIP entity shall respond with a diagnostic power mode information response within A_DoIP_Ctrl (see Table 12) after having received a previous diagnostic power mode information request.	

Table 9 — Diagnostic power mode information response

Item	Pos.	Len.	Description	Values	Support
Diagnostic power mode	0	1	Identifies whether or not the vehicle is in diagnostic power mode and ready to perform reliable diagnostics.	00_{16}: not ready 01_{16}: ready 02_{16}: not supported 03_{16} to FF_{16}: reserved by this document	mandatory

7.6 APP DoIP entity status information request and response

This payload type serves the purpose of identifying certain operating conditions of the responding DoIP entity. This allows, for example a client DoIP entity to detect existing diagnostic communication sessions as well as the capabilities of a DoIP entity.

REQ	8.DoIP-119 APP – DoIP entity status request
If supported, a DoIP entity shall implement the DoIP entity status request as specified in Table 10.	

Table 10 — DoIP entity status request

Item	Pos.	Len.	Description	Values	Support
No additional message element					

REQ	8.DoIP-120 APP – DoIP entity status response
If supported, a DoIP entity shall implement the DoIP entity status response as specified in Table 11.	

REQ	8.DoIP-121 APP – DoIP entity status response within A_DoIP_Ctrl
If supported, a DoIP entity shall respond with a DoIP entity status response within A_DoIP_Ctrl (see Table 12) after having received a previous DoIP entity status request.	

Table 11 — DoIP entity status response

Item	Pos.	Len.	Description	Values	Support
Node type (NT)	0	1	Identifies whether the contacted DoIP instance is either a DoIP node or a DoIP gateway.	00_{16}: DoIP gateway 01_{16}: DoIP node 02_{16} to FF_{16}: reserved by this document	mandatory
Max. concurrent TCP_DATA sockets (MCTS)	1	1	Represents the maximum number of concurrent TCP_DATA sockets allowed with this DoIP entity, excluding the reserve socket required for socket handling.	1 to 255	mandatory
Currently open TCP_DATA sockets (NCTS)	2	1	Number of currently established sockets.	0 to 255	mandatory
Max. data size (MDS)	3	4	Maximum size of one logical request that this DoIP entity can process.	0 to 4 GB	optional

7.7 APP timing and communication parameters

Table 12 specifies the DoIP-specific communication parameters including timeout values and payload type-specific performance requirements. In addition, the diagnostic protocol session layer timings are mapped onto the DoIP messages.

Table 12 — DoIP timing and communication parameters

Timing parameter	Description	Parameter value
A_DoIP_Ctrl	This timeout specifies the maximum time that the client DoIP entity waits for a response to a previously sent UDP message. This includes the maximum time to wait and collect multiple responses to a previous broadcast (UDP only).	Timeout: 2 s
A_DoIP_Announce_Wait	This timing parameter specifies the initial time that a DoIP entity waits until it responds to a vehicle identification request and the time that a DoIP entity waits until it transmits a vehicle announcement message after a valid IP address is configured. The value of this timing parameter shall be determined randomly between the minimum and the maximum value.	Random time: 0 to 500 ms
A_DoIP_Announce_Interval	This timing parameter specifies the time between the vehicle announcement messages that are sent by the DoIP entities after a valid IP address has been configured.	Delay time: 500 ms
A_DoIP_Announce_Num	This parameter specifies the number of vehicle announcement messages, which are sent by the DoIP entity, after the configuration of a valid IP address.	Repetition: 3 times
A_DoIP_Diagnostic_Message	This is the time between receipt of the last byte of a DoIP diagnostic message and the transmission of the confirmation ACK or NACK. After the timeout has elapsed, the request or the response shall be considered lost and the request may be repeated.	ECU performance response time: 50 ms Timeout: 2 s
T_TCP_General_Inactivity	This timeout specifies the maximum time of inactivity on a TCP_DATA socket (no data received or sent) before it is closed by the DoIP entity.	Timeout: 5 min
T_TCP_Initial_Inactivity	This timeout specifies the maximum time of inactivity directly after a TCP_DATA socket is established. After the specified time without routing activation, the TCP_DATA socket is closed by the DoIP entity.	Timeout: 2 s
T_TCP_Alive_Check	This timeout specifies the maximum time that a DoIP entity waits for an alive check response after having written an alive check request on the TCP_DATA socket. Thus, the timer elapses if the underlying TCP stack is unable to deliver the alive check request message.	Timeout: 500 ms
A_Processing_Time	This timeout is defined as the time between transmission from the client DoIP entity of DoIP messages that require no response message but may need some time to be processed. Thus, the client DoIP entity shall wait for at least A_Processing_Time before sending another request to the same DoIP entity.	Timeout: 2 s
A_Vehicle_Discovery_Timer	This is a per vehicle offboard sided timer. This timer specifies the time a vehicle can take to perform the VIN/GID synchronization between all DoIP entities. The vehicle discovery timer may only be started when a vehicle announcement/vehicle identification response message containing a VIN/GID sync status code "incomplete" (10_{16}) and a valid VIN or GID is received by the client DoIP entity.	Timeout: 5 s

7.8 APP logical addressing

Logical addresses are for diagnostic messages. A physical logical address uniquely represents a diagnostic application layer entity within any DoIP entity or on any server of the in-vehicle networks

connected via DoIP gateways. The vehicle discovery process (see 6.2) allows the client DoIP entity to map physical logical addresses to IP addresses. Functional logical addresses are used to address messages to groups of, or all of, the diagnostic application layer entities within a vehicle. As DoIP does not support multicast, for functional adressing in vehicles with multiple DoIP entities the client DoIP entity shall send unicast (point to point) IP packets to each DoIP entity, which is part of the functional address, in order to reach all servers addressed by the functional logical address. There is no mechanism to address multiple DoIP entities via a single IP address. For a DoIP gateway the reception of a functionally addressed diagnostic message implies a multi- or broadcast on the connected in-vehicle sub-networks.

Table 13 defines the addressing scheme for logical addresses.

The addressing scheme in Table 13 does not standardize individual addresses for individual servers. Thus, if a client DoIP entity wants to determine the associated functionality of a responding server DoIP entity, this needs to be carried out via other methods, for example on the application layer.

Table 13 — Logical address overview

Address	Description
0000_{16}	ISO/SAE reserved
0001_{16} to $0DFF_{16}$	VM specific
$0E00_{16}$ to $0FFF_{16}$	reserved for addresses of client
$0E00_{16}$ to $0E7F_{16}$	external legislated diagnostics test equipment (e.g. for emissions external test equipment)[a]
$0E80_{16}$ to $0EFF_{16}$	external vehicle-manufacturer-/aftermarket-enhanced diagnostics test equipment[b]
$0F00_{16}$ to $0F7F_{16}$	internal data collection/on-board diagnostic equipment (for vehicle-manufacturer use only)[c]
$0F80_{16}$ to $0FFF_{16}$	external prolonged data collection equipment (vehicle data recorders and loggers, e.g. used by insurance companies or to collect vehicle fleet data)[d]
1000_{16} to $7FFF_{16}$	VM specific
8000_{16} to $CFFF_{16}$	ISO/SAE reserved
$D000_{16}$ to $DFFF_{16}$	Reserved for SAE Truck & Bus Control and Communication Committee
$E000_{16}$ to $E3FF_{16}$	Definition of logical address is specified in use case-specific standard (e.g. ISO 27145-1, ISO 20730-1).
$E400_{16}$ to $EFFF_{16}$	vehicle-manufacturer-defined functional group logical addresses
$F000_{16}$ to $FFFF_{16}$	ISO/SAE reserved

[a] When using these addresses in the routing activation request other ongoing diagnostic communication in the vehicle may be interrupted and other normal functionality may be impaired (e.g. return to a failsafe behaviour).

[b] When using these addresses in the routing activation request and diagnostic messages the routing activation may be delayed initially due to other ongoing diagnostic communication, which may then be interrupted and other normal functionality may also be impaired (e.g. return to a failsafe behaviour).

[c] These addresses should not be used by client DoIP entity that is not designed as an integral part of the vehicle. This includes any plug-in equipment that performs diagnostic communication through the diagnostic connector.

[d] These addresses should be used by equipment that is installed in the vehicle and remains in the vehicle for periodic data retrieval by means of diagnostic communication. The DoIP entities may deny/delay accepting a routing activation request from this type of equipment in order to complete ongoing vehicle internal communication to avoid that normal operation of the vehicle may be impaired.

7.9 APP communication environments and recommended timings

Depending on the IP network scenario, different timings and influences on communication performance apply. These need to be considered when defining timing parameters for the network scenarios. This document does not specify specific timings and network setups for possible network architectures and structures.

7.10 APP DoIP entity functional requirements

REQ	8.DoIP-097 APP – DoIP GW routing diagnostic messages from client DoIP entity to server DoIP entity
Each DoIP gateway shall route user data from diagnostic messages (see Table 21) received through the TCP_DATA socket to the corresponding server DoIP entitiy on the vehicle network according to the address information contained in the diagnostic message, using the server-specific vehicle network transport protocol.	

REQ	8.DoIP-098 APP – DoIP GW routing diagnostic messages from server to client DoIP entity
Each DoIP gateway shall route the user data from the transport protocol transfer from servers on the vehicle network to the TCP_DATA socket using diagnostic messages (see Table 21) and the server-associated address information (source and target addresses).	

This implies that a DoIP gateway needs to ensure that the correct address information is used for diagnostic messages which are sent on the corresponding TCP_DATA sockets.

8 Service interface

8.1 General

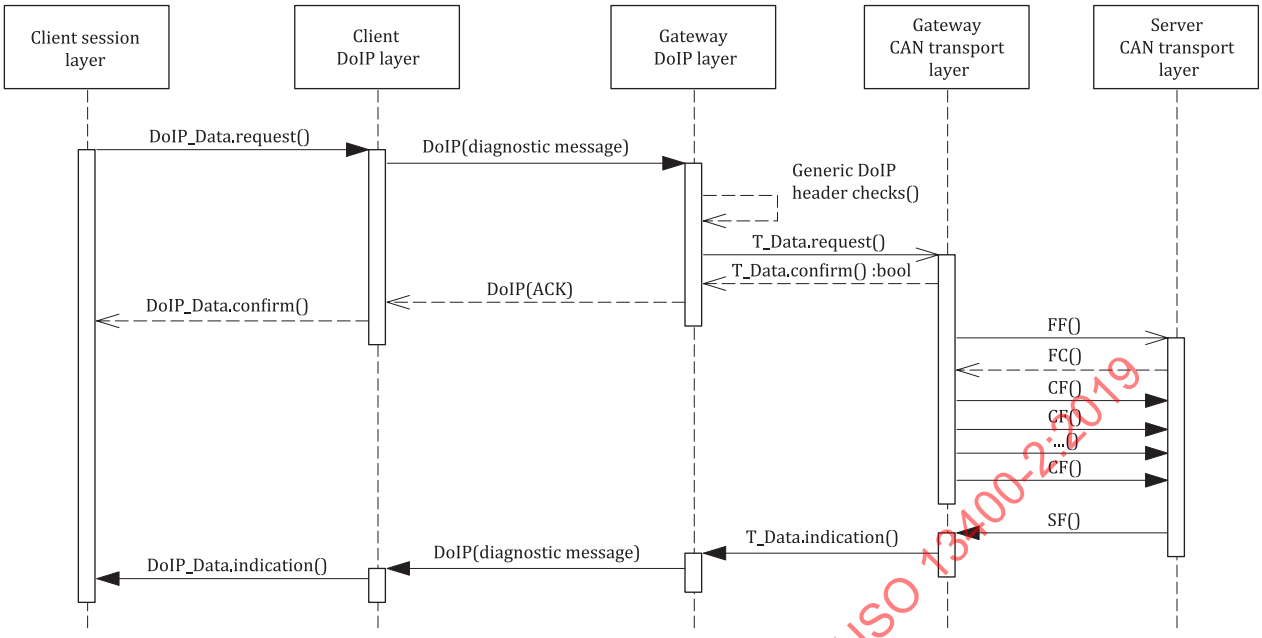
All transport layer services have the same general structure. To define the services, three types of service primitive are specified:

— a *service request primitive*, used by higher communication layers or the application to pass control information and data required to be transmitted to the network layer;

— a *service indication primitive*, used by the DoIP layer to pass status information and received data to upper communication layers or the application;

— a *service confirmation primitive*, used by the DoIP layer to pass status information to higher communication layers or the application.

This service specification does not specify an application programming interface, but only a set of service primitives that are independent of any implementation. An example of their occurrence during diagnostic communication is shown in Figure 14.



Key

CF consecutive frame
FC flow control
FF first frame
SF single frame

Figure 14 — DoIP layer service interface

All DoIP layer services have the same general format. Service primitives are written in the form:

service_name.type (
parameter A,
parameter B,
[,parameter C, ...]
)

where "service_name" is the name of the service, e.g. _DoIP_Data; "type" indicates the type of the service primitive; and "parameter A, parameter B, [parameter C, ...]" are the DoIP_SDU as a list of values passed by the service primitive. The brackets indicate that this part of the parameter list may be empty.

The service primitives define how a service user (e.g. diagnostic application) cooperates with a service provider (e.g. DoIP layer). The following service primitives are specified in this document: request, indication and confirm.

— Using the service primitive *request* (service_name.request), a service user requests a service from the service provider.

— Using the service primitive *indication* (service_name.indication), the service provider informs a service user about an internal event of the network layer or the service request of a peer protocol layer entity service user.

— With the service primitive *confirm* (service_name.confirm), the service provider informs the service user about the result of a preceding service request of the service user.

NOTE The order of parameters provided for each service primitive does not represent the order of the representing data elements in the corresponding messages, but only provides a syntactical description.

8.2 Service primitive parameters (SPP)

8.2.1 SPP data type definitions

REQ	0.DoIP-182 SPP – SPP data type definitions
The data types shall be in accordance to: — `Enum` = 8-bit enumeration — `Unsigned Byte` = 8-bit unsigned numeric value — `Unsigned Word` = 16-bit unsigned numeric value — `Unsigned Long` = 32-bit unsigned numeric value — `Byte Array` = sequence of 8-bit aligned data	

8.2.2 SPP DoIP_AI, address information

8.2.2.1 SPP DoIP_AI description

The DoIP_AI parameters are used to identify the source address (`DoIP_SA`) and target address (`DoIP_TA`) of message senders and recipients as well as the communication model for the message (`DoIP_TAtype`).

8.2.2.2 SPP DoIP_SA, DoIP logical source address

REQ	0.DoIP-183 SPP – SPP DoIP_SA, DoIP logical source address
The `DoIP_SA` parameter shall be of data type `Unsigned Word` and shall be used to encode the sending DoIP layer protocol entity. Range: [0000_{16} to $FFFF_{16}$]	

8.2.2.3 SPP DoIP_TA, DoIP logical target address

REQ	0.DoIP- 184 SPP – SPP DoIP_TA, DoIP logical target address
The `DoIP_TA` parameter shall be of data type `Unsigned Word` and shall be used to encode the receiving DoIP layer protocol entity. Range: [0000_{16} to $FFFF_{16}$]	

8.2.2.4 SPP DoIP_TAtype, DoIP logical target address type

The parameter DoIP_TAtype is an extension to the DoIP_TA parameter.

REQ	0.DoIP- 185 SPP – SPP DoIP_TA, DoIP logical target address
The DoIP_TAtype parameter shall be of data type Enum and shall be used to encode the communication model used by the communicating peer entities of the DoIP layer. Two communication models shall be supported: — 1-to-1 communication, called physical addressing (unicast), and — 1-to-n communication, called functional addressing (multicast/broadcast). Range: [00_{16} to FF_{16}]	

NOTE See 7.8 for details on mapping physical and functional logical addresses to IP addresses.

8.2.3 SPP Length, length of PDU

REQ	0.DoIP- 186 SPP – Length, length of PDU
The Length parameter shall be of data type Unsigned Long and shall be used to encode the length of data to be transmitted/received. Range: [$0000\ 0000_{16}$ to $FFFF\ FFFF_{16}$] (0 GB to 4 GB (2^{32} bytes))	

8.2.4 SPP PDU, protocol data unit

REQ	0.DoIP- 187 SPP – PDU, protocol data unit
The PDU parameter shall be of data type Byte Array and shall contain the message data content that the higher layer entities exchange. Range: [00_{16} to FF_{16}]	

8.2.5 SPP DoIP_Result

REQ	0.DoIP- 188 SPP – DoIP_Result
The DoIP_Result parameter shall be of data type Enum and shall contain the status relating to the outcome of a service execution. If two or more errors are discovered at the same time, then the network layer entity shall use the parameter value first found in this list in the error indication to the higher layers. Range: [DoIP_OK, DoIP_HDR_ERROR, DoIP_TIMEOUT_A, DoIP_UNKNOWN_SA, DoIP_INVALID_SA, DoIP_UNKNOWN_TA, DoIP_MESSAGE_TOO_LARGE, DoIP_OUT_OF_MEMORY, DoIP_TARGET_UNREACHABLE, DoIP_NO_LINK, DoIP_NO_SOCKET, DoIP_ERROR]	

NOTE DoIP_Results are defined by the DoIP diagnostic message handler logic in Figure 17.

8.3 SPP DoIP layer service interface

8.3.1 SPP DoIP_Data.request

The DoIP_Data.request service requests transmission of <PDU> with <Length> bytes from the sender to the receiver peer entities identified by the address information in DoIP_SA, DoIP_TA and DoIP_TAtype (see 8.2 for service primitive parameter definition).

Each time the DoIP_Data.request service is called, the DoIP layer shall signal the completion (or failure) of the message transmission to the service user by issuing a DoIP_Data.confirm service call:

DoIP_Data.request (
DoIP_SA
DoIP_TA
DoIP_TAtype
<PDU>
<Length>
)

8.3.2 SPP DoIP_Data.confirm

The DoIP_Data.confirm service is issued by the DoIP layer. The service primitive confirms the completion of a DoIP_Data.request service identified by the address information in DoIP_SA, DoIP_TA and DoIP_TAtype. The parameter <DoIP_Result> provides the status of the service request (see 8.2 for parameter definition).

DoIP_Data.confirm (
DoIP_SA
DoIP_TA
DoIP_TAtype
<DoIP_Result>
)

8.3.3 SPP DoIP_Data.indication

The DoIP_Data.indication service is issued by the DoIP layer. The service primitive indicates <DoIP_Result> events and delivers <PDU> with <Length> bytes received from a peer protocol entity identified by the address information in DoIP_SA, DoIP_TA and DoIP_TAtype to the adjacent upper layer (see 8.2 for parameter definition).

The parameters <PDU> and <Length> are only valid if <DoIP_Result> equals DoIP_OK.

DoIP_Data.indication (
DoIP_SA
DoIP_TA
DoIP_TAtype
<PDU>
<Length>
<DoIP_Result>
)

The DoIP_Data.indication service call is issued after the reception of a DoIP diagnostic message.

If the DoIP layer detects any type of error in a DoIP diagnostic message, then the message shall be ignored by the DoIP layer and no DoIP_Data.indication shall be issued to the adjacent upper layer.

9 Application layer (AL)

9.1 AL dynamic host control protocol (DHCP)

9.1.1 AL general

The dynamic host configuration protocol is a client-server DoIP entity networking protocol that provides a mechanism for allocation of IP addresses. DHCP provides the mechanism for a dynamically configured client DoIP entity to acquire all of the IP configuration parameters that it needs in order to

communicate successfully over the local network. DHCP is an application layer protocol in accordance with the OSI layered architecture model (see Table 14).

REQ	7.DoIP-101 AL – Single DHCP server DoIP entity
It shall be ensured that none of the DoIP entities provide DHCP server DoIP entity services on the link that is connected to the client DoIP entity, in order to avoid disturbing the client DoIP entity network (e.g. sending DHCP_OFFER messages and providing different IP gateway and DNS server DoIP entity addresses).	

Table 14 — DHCP on OSI layers

OSI layer	Protocol
Application	IPv4: DHCP (IETF RFC 2131) IPv6: DHCPv6 (IETF RFC 3315)
Transport	UDP
Network	IP (IPv4, IPv6)
Data link / Physical	Ethernet (ISO/IEC/IEEE 8802-3)

REQ	7.DoIP-014 AL – IPv4 DHCP client DoIP entity
If IPv4 is used, each DoIP entity shall implement the DHCP client DoIP entity behaviour as specified in IETF RFC 2131.	

REQ	7.DoIP-015 AL – IPv6 DHCPv6 client DoIP entity
If IPv6 is used, each DoIP entity shall implement the DHCPv6 client DoIP entity behaviour as specified in IETF RFC 3315.	

NOTE 1 When supporting either DoIP-014 or DoIP-015, considering DoIP-109 is important.

REQ	7.DoIP-016 AL – DoIP entity host name option
Each DoIP entity shall implement either the “host name option” as defined in IETF RFC 2132, or the “fully qualified domain name” as defined in IETF RFC 4702.	

REQ	7.DoIP-017 AL – Host name option DoIP-<manufacturer_specific>
The host name option shall contain at minimum “DoIP-<manufacturer_specific>”, where the <manufacturer_specific> part can be replaced with any text that meets the specific requirements of the manufacturer.	

NOTE 2 The host name option is used to allow for detection of a DoIP-compliant vehicle in a network which currently uses a DHCP-assigned IP address. An example of the implementation of requirement DoIP-017 is the host name option “DoIP-VIN12345678901234567” or simply “DoIP-” if the manufacturer-specific part is left empty.

REQ	7.DoIP-138 AL – DoIP entity IP address assignment
If the DoIP activation line is actived as specified in ISO 13400-3 each DoIP entity shall start the IP address assignment process for its externally accessible interfaces.	

NOTE 3 Additional mechanisms might be required for vehicle network architectures containing more than one DoIP entity, in order to ensure that all DoIP entities start assigning a valid IP address once the activation line is activated.

9.1.2 AL IP address assignment

9.1.2.1 AL general

This subclause specifies how a DoIP entity acquires a valid IP address in order to communicate over an IP-based network. In general, the following parameters are needed for IP addressing:

— IP address (IPv4, IPv6);

— subnet mask (IPv4 only);

— prefix length (IPv6 only).

If the DoIP entity is integrated into a network infrastructure, the additional parameter default gateway address (= IP address of the default router) (IPv4, IPv6) is needed.

The network infrastructure provides dynamically assigned IP addresses or requires the DoIP entity to independently assign an IP address, which does not conflict with the IP addresses of other nodes on the local network.

9.1.2.2 AL IPv4 address assignment

Depending on whether a DoIP entity is in an infrastructure environment or whether it operates in a direct peer to peer connection, IP addresses need to be assigned taking into consideration the specifics of IPv4. This subclause describes how an IP address can be configured in a minimum of time, to ensure fast connection establishment.

REQ	**7.DoIP-099 AL – DoIP entity dynamic configuration of IPv4 link-local addresses**
Each DoIP entity shall implement the dynamic configuration of IPv4 link-local addresses as specified in IETF RFC 3927 for its externally accessible interfaces.	

To speed up the process of assigning an IP address on a direct peer-to-peer connection, it is recommended that the values listed in Table 15 be used by the DoIP entity when verifying the link-local IP address as specified in IETF RFC 3927. In the best-case scenario, when using the values from Table 15, the DoIP entity has an IP address configured in approximately two seconds. For client DoIP entity using the performance values recommended by this document (listed in Table 15), in the best-case scenario an IP address is configured after approximately seven seconds.

IMPORTANT — If the client DoIP entity is based on standard operating systems, the overall time to acquire an IP address depends on the configuration and the IP address assignment algorithms of these operation systems and may range from several seconds up to minutes.

Table 15 defines the IETF RFC 3927 adapted timings.

Table 15 — IETF RFC 3927 adapted timings

Parameter	This document's **recommended performance values**	**IETF RFC 3927 value**	**Description/rationale**
PROBE_WAIT	1 s	1 s	Time before first probe message after link becomes active (initial delay)
PROBE_NUM	1 message	3 messages	Number of probe messages
PROBE_MIN	1 s	1 s	Minimum time between ARP probe messages
PROBE_MAX	1 s	2 s	Maximum time between ARP probe messages
ANNOUNCE_WAIT	1 s	2 s	Delay before announcing locally configured IP address

Table 15 *(continued)*

Parameter	This document's **recommended performance values**	**IETF RFC 3927 value**	**Description/rationale**
ANNOUNCE_NUM	1 message	2 messages	Number of announcement messages
ANNOUNCE_INTERVAL	1 s	2 s	Time between announcement messages

REQ	**7.DoIP-018 AL – IPv4 DoIP entity concurrent AutoIP-based and DHCP-assigned IP address**
For improved IPv4 address-assignment performance, each DoIP entity shall perform the AutoIP-based and DHCP-assigned IP address assignment concurrently, as specified in Figure 15, when a data-link connection is detected (for the DoIP edge node) or when remotely invoked.	

REQ	**7.DoIP-019 AL – DoIP entity AutoIP- or a DHCP-based IP address**
Each DoIP entity shall configure either an AutoIP-based IP address or a DHCP-based IP address, depending on which IP address assignment results in a valid IP address first.	

REQ	**7.DoIP-020 AL – DHCP- supersede AutoIP-assigned IP addresses**
DHCP-assigned IP addresses supersede AutoIP-assigned IP addresses, implying that the reception of a DHCP-assigned IP address shall overwrite any previously configured link-local IP address (deviation from IETF RFC 3927).	

REQ	**7.DoIP-021 AL – DoIP entity first DHCP_OFFER message**
Each DoIP entity shall use the first DHCP_OFFER message with an IP other than 0.0.0.0 to configure a DHCP-assigned IP address.	

REQ	**7.DoIP-023 AL – DoIP entity restart (DHCP_DISCOVER)**
Each DoIP entity shall restart (DHCP_DISCOVER) the attempt to configure a DHCP-assigned IP address if no valid DHCP-configured IP address could be configured after a total time of 10 s.	

DoIP-023 defines an overall timeout for triggering the re-start of the DHCP-based IP address assignment. The value chosen (10 s), differs from the recommended retransmission logic in the related RFC to allow for faster IP address assignment when using DHCP. This document does not specify the timing of the individual steps of the DHCP process. It is the VM's responsibility to specify the timing and retry requirements needed to meet the overall time requirement of DoIP-023.

Figure 15 shows the concurrent IPv4 AutoIP-based and DHCP-based IP address configuration.

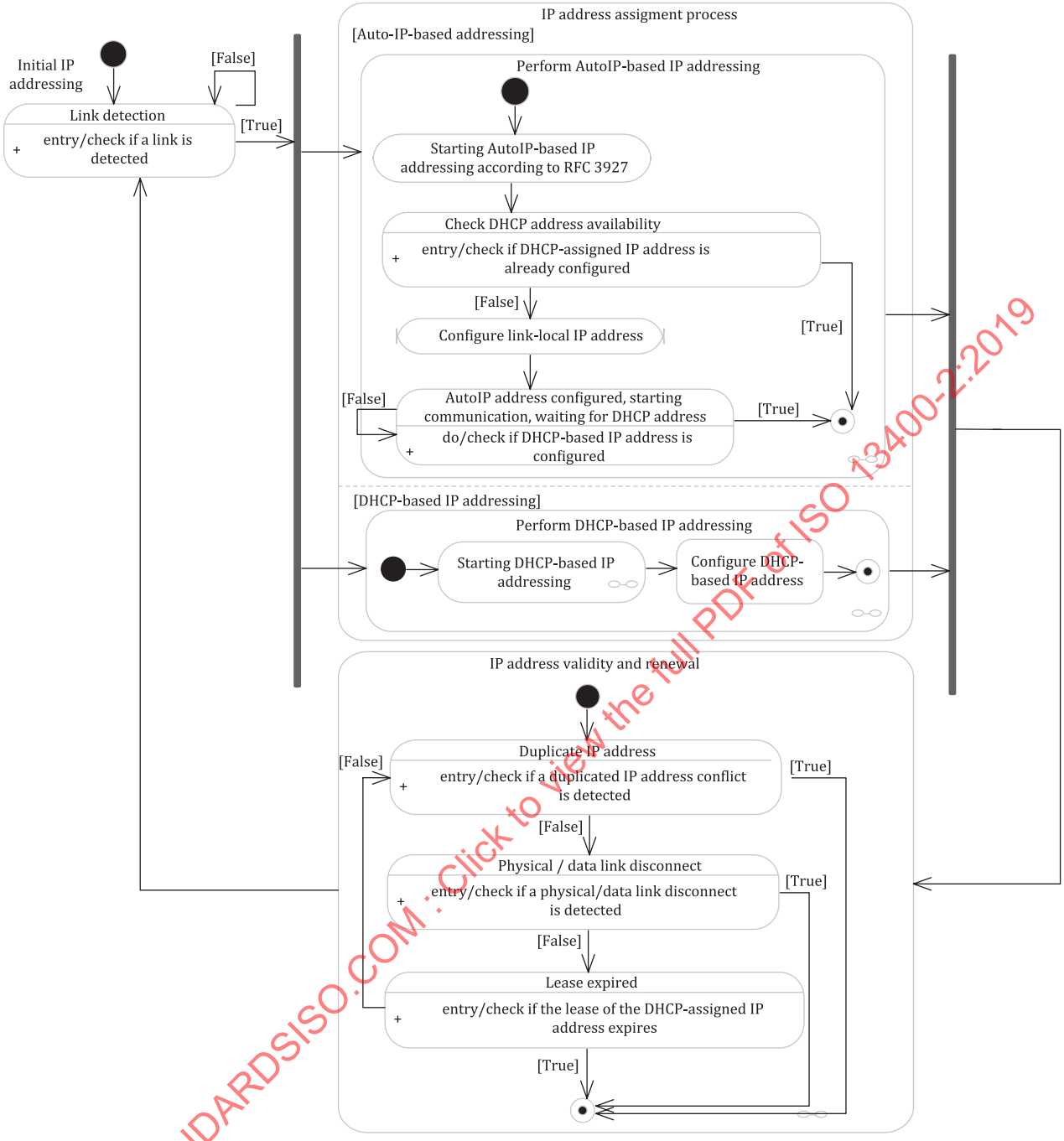


Figure 15 — Concurrent IPv4 AutoIP-based and DHCP-based IP address configuration

9.1.2.3 AL IPv6 address assignment

Due to the capabilities of the IPv6 protocol, the IP address assignment process differs slightly from the IPv4-based process. Specifically, the assignment of an IPv6 address for a direct peer-to-peer connection is considerably simpler and faster than the AutoIP-based address assignment of IPv4 as it uses the hardware address of the network interface.

NOTE An IPv6 host usually has multiple IP addresses assigned to a physical network interface. The rules for prioritization of the different IPv6 addresses are specified in the referenced RFCs.

REQ	7.DoIP-024 AL – IPv6 DoIP entity link-local IPv6 unicast address
For IPv6, each DoIP entity shall support the configuration of a link-local IPv6 unicast address as specified in IETF RFC 4291 for its externally accessible interfaces.	

REQ	7.DoIP-025 AL – DoIP entity interface ID from IEEE 48 bit MAC identifier
The interface ID of the link-local address of a DoIP entity shall be generated from its IEEE 48 bit MAC identifier as defined in IETF RFC 4291.	

REQ	7.DoIP-139 AL – IPv6 address from router advertisement messages
If present in the network, an IPv6 address shall be derived from the router advertisement messages in accordance with IETF RFC 4862.	

REQ	7.DoIP-140 AL – IPv6 address gathered from DHCPv6 server DoIP entity
If present in the network, an IPv6 address shall be gathered from a DHCPv6 server DoIP entity.	

REQ	7.DoIP-141 AL – IPv6 source address selection
IPv6 source address selection shall be performed in accordance with IETF RFC 3484.	

9.1.3 AL IP address validity and renewal

This subclause specifies the requirements for when to discard a configured IP address and how to renew it. The IP address discarding and renewal process and criteria are depicted in Figure 18.

REQ	7.DoIP-028 AL – DoIP entity discard IP address
Each DoIP entity shall discard its IP dynamically assigned address when any one of the following conditions occurs: — the lease of the DHCP-assigned IP address expires, including DHCPNAK messages or a new DHCP-assigned IP address is received (for further details, see IETF RFC 2131 for IPv4 and IETF RFC 4291 for IPv6); — the client DoIP entity disconnects (if the physical layer can detect this event) or the activation line is in the "deactivation criteria fulfilled" state; — a duplicate IP address conflict is detected; — the IP address is remotely invalidated (optional; vehicle-manufacturer-specific implementation).	

REQ	7.DoIP-029 AL – DoIP entity configure a new IP address
Each DoIP entity shall attempt to configure a new IP address as specified in 9.1.2.2 for IPv4 or 9.1.2.3 for IPv6 if it has discarded its IP address due to one of the reasons in requirement DoIP-028.	

If a DHCP-assigned IP address is available and the lease time hasn't expired a DoIP node may verify and reuse (if the IP address is still available) its previously allocated IP address (see IETF RFC 2131). This applies to requirements [DoIP-028] and [DoIP-029].

REQ	7.DoIP-030 AL – DoIP entity close and reset all TCP sockets
Each DoIP entity shall close and reset all TCP sockets, including any authentications (e.g. routing activation) on these sockets, when the underlying IP stack discards or invalidates its current IP address.	

9.2 AL generic DoIP protocol message structure

A generic DoIP protocol message structure means, that all messages, which are sent or received over TCP_DATA or UDP_TEST_EQUIPMENT_REQUEST or UDP_DISCOVERY, contain the generic header specified in Table 16. The generic DoIP header is processed and negatively acknowledged as depicted in Figure 15.

REQ	7.DoIP-036 AL – DoIP entity implements generic DoIP header structure
Each DoIP entity shall implement the generic DoIP header structure for all DoIP messages, as specified in Table 16. This part of the message is located at the beginning of each DoIP message.	

REQ	7.DoIP-156 AL – DoIP entity supports protocol version
Each DoIP entity shall support the protocol version default value for vehicle identification request messages as specified in Table 16. This means that a DoIP entity shall always ignore the protocol version default value in vehicle identification request messages.	

EXAMPLE If client DoIP entity supports multiple protocol versions at the same time, and there is no information regarding the DoIP versions supported by the DoIP entities, this default value is used by the test equipment in the vehicle identification request messages.

NOTE 1 For UDP datagram-based messages, this implies that the generic header is located in the first bytes of the payload. For TCP-based data, the header separates the individual DoIP messages within the data stream.

The generic DoIP header uses four bytes to encode the payload size, limiting the number of bytes in the payload to 4 GB (4 294 967 295 bytes).

NOTE 2 The maximum allowed payload length is limited by the specific transport layer that is used in the vehicle.

NOTE 3 For easier understanding, debugging and optimized implementations, DoIP payload types are grouped according to their message contents. Groups are node management ($XX16_{16}$), vehicle information ($4XXX_{16}$) and diagnostics ($8XXX_{16}$).

REQ	7.DoIP-037 AL – DoIP entity processes generic DoIP header structure
Each DoIP entity shall process the generic DoIP header structure for all DoIP messages in the order specified in Figure 15.	

Table 16 defines the generic DoIP header structure.

Table 16 — Generic DoIP header structure

Item	Pos.	Len.	Description	Values
Protocol version	0	1	Identifies the protocol version of DoIP packets.	00_{16}: reserved 01_{16}: ISO/DIS 13400-2:2010 02_{16}: ISO 13400-2:2012 03_{16}: this document 04_{16} to FE_{16}: reserved by this document FF_{16}: default value for vehicle identification request messages
Inverse protocol version	1	1	Contains the bit-wise inverse value of the protocol version, which is used in conjunction with the DoIP protocol version as a protocol verification pattern to ensure that a correctly formatted DoIP message is received.	Equals the <Protocol_Version> XOR FF_{16} (e.g. FE_{16} for protocol version 01_{16}).

Table 16 *(continued)*

Item	Pos.	Len.	Description	Values
Payload type (GH_PT)	2	2	Contains information about how to interpret the data following the generic DoIP header (e.g. gateway command, diagnostic message, etc.)	See Table 17 for a complete list of currently specified payload type values.
Payload length (GH_PL)	4	4	Contains the length of the DoIP message payload in bytes (i.e. excluding the generic DoIP header bytes). Some payload types do not require any additional parameters (payload length is 0), some require a fixed DoIP message length while others allow for dynamic length DoIP messages.	0 to 4 294 967 295 bytes (= <d>)
Payload type specific message content	8	…	The payload type specific message content starts here. This implies that, for example, byte position 0 of the payload type-specific part of the message (see 12.5.1) means byte position 8 in the context of the overall DoIP message.	—

Table 17 provides an overview of the payload types that are specified for DoIP.

Table 17 — Overview of DoIP payload types

Payload type value	Payload type name	Specified in subclause	Support (DoIP gateways)	Support (DoIP nodes)	Port and protocol
0000_{16}	Generic DoIP header negative acknowledge	9.3	mandatory	mandatory	UDP_DISCOVERY UDP_TEST_EQUIPMENT_REQUEST TCP_DATA
0001_{16}	Vehicle identification request message	7.4	mandatory	mandatory	UDP_DISCOVERY
0002_{16}	Vehicle identification request message with EID	7.4	optional	optional	UDP_DISCOVERY
0003_{16}	Vehicle identification request message with VIN	7.4	mandatory	mandatory	UDP_DISCOVERY
0004_{16}	Vehicle announcement message/vehicle identification response message	7.4	mandatory	mandatory	UDP_DISCOVERY UDP_TEST_EQUIPMENT_REQUEST
0005_{16}	Routing activation request	12.5.2	mandatory	mandatory	TCP_DATA
0006_{16}	Routing activation response	12.5.2	mandatory	mandatory	TCP_DATA
0007_{16}	Alive check request	9.6	mandatory	mandatory	TCP_DATA
0008_{16}	Alive check response	9.6	mandatory	mandatory	TCP_DATA
0009_{16} to 4000_{16}	Reserved by this document				
4001_{16}	DoIP entity status request	7.6	optional	optional	UDP_DISCOVERY

Table 17 *(continued)*

Payload type value	Payload type name	Specified in subclause	Support (DoIP gateways)	Support (DoIP nodes)	Port and protocol
4002_{16}	DoIP entity status response	7.6	optional	optional	UDP_TEST_EQUIPMENT_REQUEST
4003_{16}	Diagnostic power mode information request	7.5	mandatory	mandatory	UDP_DISCOVERY
4004_{16}	Diagnostic power mode information response	7.5	mandatory	mandatory	UDP_TEST_EQUIPMENT_REQUEST
4005_{16} to 8000_{16}	Reserved by this document				
8001_{16}	Diagnostic message	9.5	mandatory	mandatory	TCP_DATA
8002_{16}	Diagnostic message positive acknowledgement	9.5	mandatory	mandatory	TCP_DATA
8003_{16}	Diagnostic message negative acknowledgement	9.5	mandatory	mandatory	TCP_DATA
8004_{16} to $EFFF_{16}$	Reserved by this document				
$F000_{16}$ to $FFFF_{16}$	Reserved for manufacturer-specific use	—	optional	optional	—

Figure 16 shows the DoIP generic header handler.

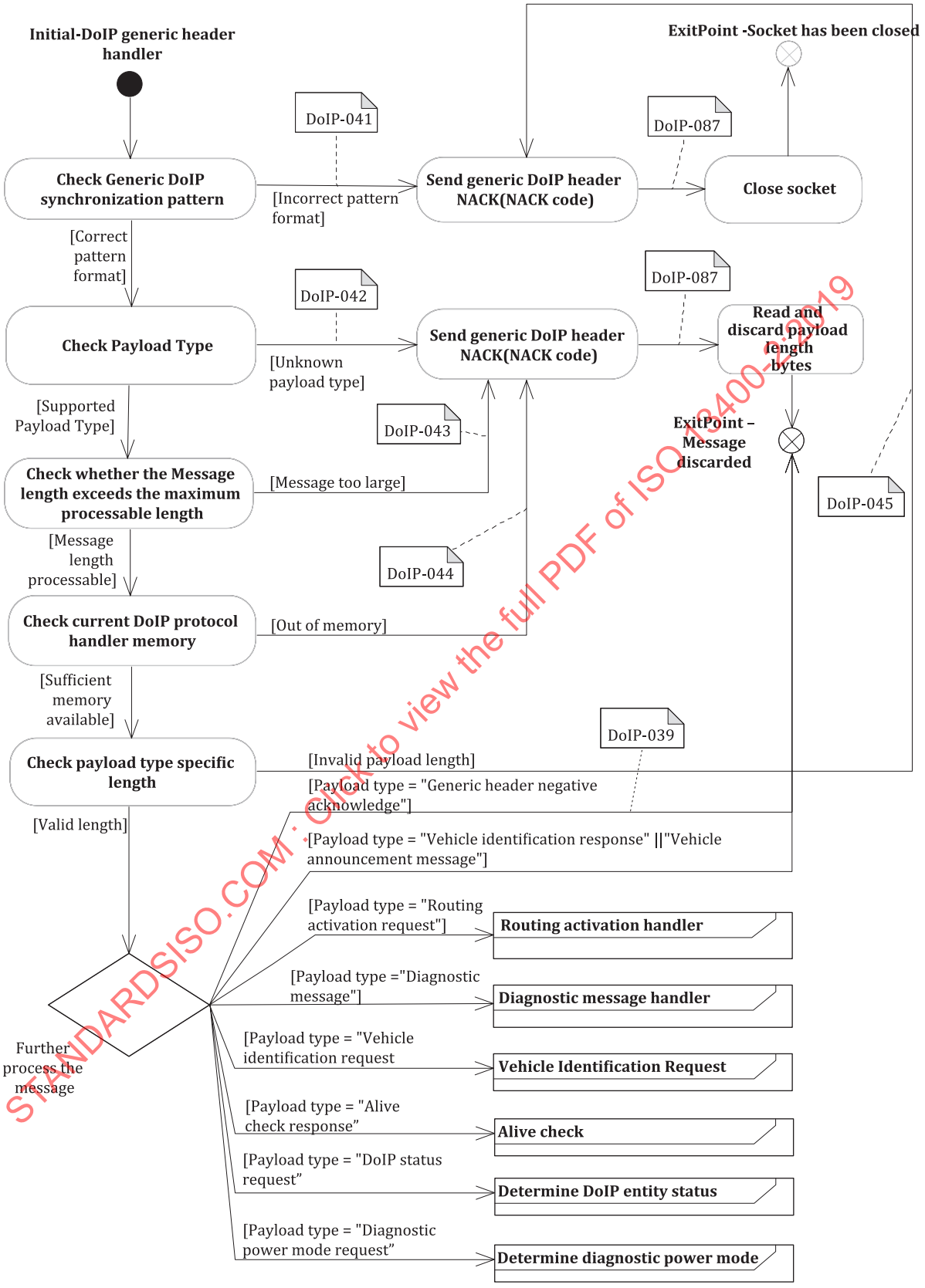


Figure 16 — DoIP generic header handler

As generic header negative acknowledge messages are also DoIP messages, they need to include the generic header part as specified in Table 16.

REQ	7.DoIP-038 AL – DoIP entity supports generic DoIP header NACK structure
Each DoIP entity shall support the generic DoIP header negative acknowledge structure as specified in Table 18.	

REQ	7.DoIP-087 AL – DoIP entity supports generic DoIP header NACK codes
Each DoIP entity shall perform the required action specified in Table 19 after having sent the generic DoIP header negative acknowledge message.	

REQ	7.DoIP-039 AL – DoIP entity ignores generic DoIP header NACK messages
Each DoIP entity shall ignore received generic DoIP header negative acknowledge messages.	

REQ	7.DoIP-040 AL – Client DoIP entity generic DoIP header NACK messages
The client DoIP entity shall not send generic DoIP header negative acknowledge messages upon receipt of an incorrect DoIP message from a DoIP entity.	

The generic DoIP header negative acknowledge message may only be used for determining the error condition for a previously sent DoIP message.

NOTE A client DoIP entity can use DoIP header negative acknowledge messages during the development phase to verify correct implementation of DoIP messages in a DoIP entity.

Table 18 defines the generic DoIP header negative acknowledge structure.

Table 18 — Generic DoIP header negative acknowledge structure

Item	Pos.	Len.	Description	Values
Generic DoIP header NACK code	0	1	The generic header negative acknowledge code indicates the specific error, detected in the generic DoIP header, or it indicates an unsupported payload or a memory overload condition.	See Table 19.

Table 19 defines the generic DoIP header NACK codes.

Table 19 — Generic DoIP header NACK codes

Value	Description	Required action	Support
00_{16}	incorrect pattern format	close socket	mandatory
01_{16}	unknown payload type	discard DoIP message	mandatory
02_{16}	message too large	discard DoIP message	mandatory
03_{16}	out of memory	discard DoIP message	mandatory
04_{16}	invalid payload length	close socket	mandatory
05_{16} to FF_{16}	reserved by this document	—	—

REQ	7.DoIP-041 AL – DoIP entity does not match generic DoIP header structure
Each DoIP entity shall send a generic DoIP header negative acknowledge message with NACK code set to 00_{16} if the protocol version or inverse protocol version (synchronization pattern) does not match the format specified in Table 16.	

REQ	7.DoIP-042 AL – DoIP entity NACK code set to 01_{16}
Each DoIP entity shall send a generic DoIP header negative acknowledge message with NACK code set to 01_{16} if the payload type is not supported by the DoIP entity.	

REQ	7.DoIP-043 AL – DoIP entity NACK code set to 02_{16}
Each DoIP entity shall send a generic DoIP header negative acknowledge message with NACK code set to 02_{16} if the payload length (DoIP message request without the DoIP header) exceeds the maximum DoIP maximum data size (MDS) supported by the DoIP entity regardless of the current memory utilization.	

REQ	7.DoIP-044 AL – DoIP entity NACK code set to 03_{16}
Each DoIP entity shall send a generic DoIP header negative acknowledge message with NACK code set to 03_{16} if the payload length exceeds the currently available DoIP protocol handler memory of the DoIP entity.	

REQ	7.DoIP-045 AL – DoIP entity NACK code set to 04_{16}
Each DoIP entity shall send a generic DoIP header negative acknowledge message with NACK code set to 04_{16} if the payload length parameter does not match the expected length for the specific payload type. This includes payload-type-specific minimum length, fixed length and maximum length checks.	

9.3 AL handling of UDP packets and TCP data

This subclause specifies the general requirements, which apply to the handling of UDP packets and TCP data.

REQ	7.DoIP-031 AL – Ignore packets with a multi- or broadcast address as the source IP address
Any packets with a multi- or broadcast address as the source IP address shall be ignored by the receiving DoIP entity.	

REQ	7.DoIP-122 AL – One DoIP message per UDP datagram
Only one DoIP message shall be transmitted by any DoIP entity per UDP datagram.	

9.4 AL supported payload types over TCP and UDP ports

Table 20 provides an overview of all TCP and UDP ports for unsecured and secured communication defined in this document and their intended use.

Table 20 — UDP and TCP port usage

Use case	Payload type	Sender	Source port	Receiver	Destination port	Protocol	Addressing
Vehicle discovery	Vehicle identification request	Client DoIP entity	UDP_TEST_EQUIPMENT_REQUEST	DoIP entity	UDP_DISCOVERY	UDP	Multi- or unicast
Vehicle discovery	Vehicle identification response	DoIP entity	UDP_DISCOVERY or dynamically assigned	client DoIP entity	UDP_TEST_EQUIPMENT_REQUEST	UDP	Unicast

Table 20 *(continued)*

Use case	Payload type	Sender	Source port	Receiver	Destination port	Protocol	Addressing
Vehicle discovery	Vehicle announcement	DoIP entity	UDP_DISCOVERY or dynamically assigned	client DoIP entity	UDP_DISCOVERY	UDP	Multicast
Data transmission	e.g. Routing activation request	Client DoIP entity	Dynamically assigned	DoIP entity	TCP_DATA	TCP TLS	Unicast
Data transmission	e.g. Routing activation response	DoIP entity	TCP_DATA	client DoIP entity	dynamically assigned	TCP TLS	Unicast

NOTE 1 Table 20 shows the IPv6 variant. For IPv4, Broadcast is used instead of Multicast.

NOTE 2 In case of secured communication, only the TCP connection over the dedicated TCP_DATA port 3496 for TLS is secured. For Vehicle discovery and UDP protocol based DoIP communication, no secured communication and no dedicated ports are applied.

9.5 AL diagnostic message and diagnostic message acknowledgement

This subclause specifies the message format that allows for routing of diagnostic messages (i.e. diagnostic requests) onto the vehicle networks and from the vehicle networks (i.e. diagnostic responses) back to the client DoIP entity. If the diagnostic message is sent by the client DoIP entity, DoIP entities always acknowledge (positively or negatively) these messages. Diagnostic messages can also be sent by the DoIP entities, for example when transmitting a diagnostic response or an unsolicited message (e.g. response on event) from a server DoIP entity to the client DoIP entity. In this case, the diagnostic messages is not acknowledged by the client DoIP entity.

REQ	7.DoIP-064 AL – DoIP entity diagnostic message structure
Each DoIP entity shall support the diagnostic message structure as specified in Table 21 for incoming (i.e. requests) and outgoing (i.e. responses) diagnostic messages.	

Table 21 — Payload type diagnostic message structure

Item	Pos.	Len.	Description	Values	Support condition
Source address (SA)	0	2	Contains the logical address of the sender of a diagnostic message (e.g. the client DoIP entity address).	See Table 13.	mandatory
Target address (TA)	2	2	Contains the logical address of the receiver of a diagnostic message (e.g. a specific server DoIP entity on the vehicle's networks).	See Table 13.	mandatory
User data (UD)	4	d - 4	Contains the actual diagnostic data (e.g. a ISO 14229-1 diagnostic request), which shall be routed to the destination (e.g. the ECM).	See Table 22 for an example.	mandatory

REQ	7.DoIP-065 AL – DoIP entity diagnostic messages on TCP_DATA sockets
Each DoIP entity shall receive and process diagnostic messages on its TCP_DATA sockets in the order specified in Figure 22.	

Table 22 gives an example of how an ISO 27145-3 diagnostic message is transported by a DoIP diagnostic message frame.

Table 22 — Example of ISO 27145-3 request message transported by a DoIP message frame

Message direction:	client → vehicle		
Message type:	Functionally addressed request message (read protocol identification InfoType identifier)		
Data byte	**Description**	**Byte value**	**Mnemonic**
0	The ISO 13400 series – protocol version	01_{16}	—
1	The ISO 13400 series – inverse protocol version	FE_{16}	—
2	The ISO 13400 series – payload type	8001_{16}	GH_PT
3	The ISO 13400 series – payload type		GH_PT
4	The ISO 13400 series – payload length	7	GH_PL
5	The ISO 13400 series – payload length		GH_PL
6	The ISO 13400 series – payload length		GH_PL
7	The ISO 13400 series – payload length		GH_PL
8	The ISO 13400 series – source address	e.g. $0E00_{16}$	SA
9	The ISO 13400 series – source address		SA
10	The ISO 13400 series – target address	$E000_{16}$	TA
11	The ISO 13400 series – target address		TA
12	The ISO 13400 series – user data / ISO 27145-3 – ReadDataByIdentifier request SID	22_{16}	UD / RDBI
13	The ISO 13400 series – user data / ISO 27145-3 – DataIdentifier #1 (HB) = ITID = protocol identification	$F8_{16}$	UD / DID_HB
14	The ISO 13400 series – user data / ISO 27145-3 – DataIdentifier #1 (LB) = ITID = protocol identification	10_{16}	UD / DID_LB

Figure 17 depicts the DoIP diagnostic message handler.

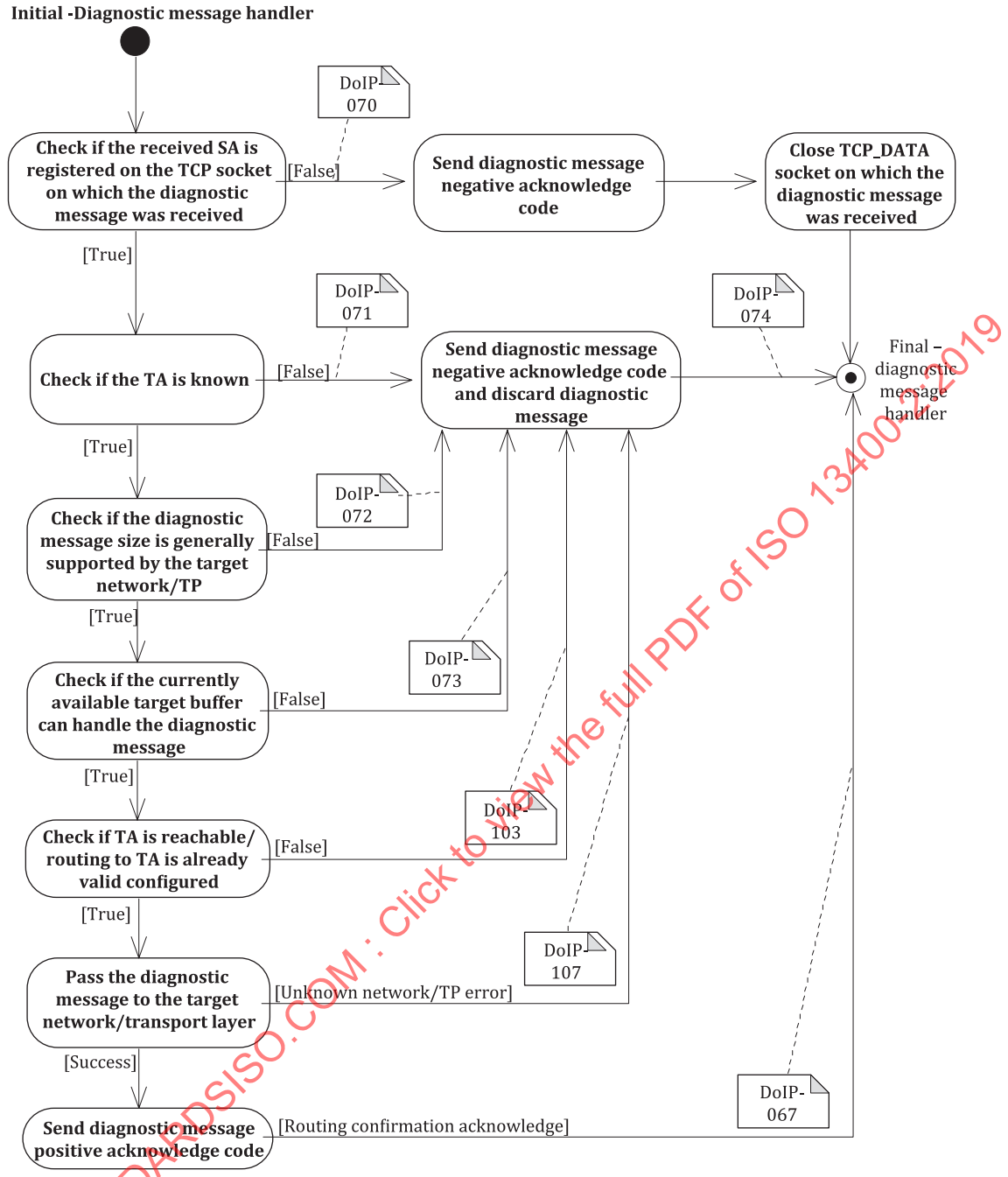


Figure 17 — DoIP diagnostic message handler

REQ	7.DoIP-066 AL – DoIP entity diagnostic message positive acknowledgement
Each DoIP entity shall support the diagnostic message positive acknowledgement as specified in Table 23.	

Table 23 — Payload type diagnostic message positive acknowledgment structure

Item	Pos.	Len.	Description	Values	Support
Source address (SA)	0	2	Contains the logical address of the (intended) receiver of the previous diagnostic message (e.g. a specific server DoIP entity on the vehicle's networks).	See Table 13.	mandatory

Table 23 *(continued)*

Item	Pos.	Len.	Description	Values	Support
Target address (TA)	2	2	Contains the logical address of the sender of the previous diagnostic message (i.e. the client DoIP entity address).	See Table 13.	mandatory
ACK code	4	1	Contains the diagnostic message positive acknowledge code.	See Table 24.	mandatory
Previous diagnostic message data	5	0 to <d - 5>	May contain a copy of up to <d - 5> bytes of the diagnostic message (max. size <d - 4>) that is currently acknowledged. This may aid in troubleshooting communication problems.	—	optional

Table 24 — Diagnostic message positive acknowledge codes

Value	Description	Support
0016	Routing confirmation acknowledge (ACK) message indicates a correctly received diagnostic message, which is processed and put into the transmission buffer of the destination network.	mandatory
0116 to FF16	Reserved by this document.	—

REQ	7.DoIP-067 AL – DoIP entity diagnostic message positive ACK code set to 00_{16}
Each DoIP entity shall send the diagnostic message positive acknowledgement with ACK code set to 00_{16} (see Table 24) immediately after the diagnostic message has been correctly processed and copied into the destination network transmission buffer.	

REQ	7.DoIP-068 AL – DoIP entity diagnostic message negative acknowledgement structure
Each DoIP entity shall support the diagnostic message negative acknowledgement as specified in Table 25.	

Table 25 — Payload type diagnostic message negative acknowledgment structure

Item	Pos.	Len.	Description	Values	Support
Source address (SA)	0	2	Contains the logical address of the (intended) receiver of the previous diagnostic message (e.g. a specific server DoIP entity on the vehicle's networks).	See Table 13.	mandatory
Target address (TA)	2	2	Contains the logical address of the sender of the previous diagnostic message (i.e. the client DoIP entity address).	See Table 13.	mandatory
NACK code	4	1	Contains the diagnostic message negative acknowledge code.	See Table 26.	mandatory
Previous diagnostic message data	5	0 to <d - 5>	May contain a copy of up to <d - 5> bytes (only limited by the maximum supported DoIP message size) of the diagnostic message that is currently acknowledged. This may aid in troubleshooting communication problems and is up to the manufacturer to select a feasible number of bytes to repeat.	HEX	optional

REQ	7.DoIP-070 AL – DoIP entity diagnostic message NACK code set to 02_{16}
Each DoIP entity shall send the diagnostic message negative acknowledgement with NACK code set to 02_{16} (see Table 26) and close the TCP_DATA socket when the diagnostic message contains a source address which is not activated on the TCP_DATA socket on which the diagnostic message is received.	

REQ	7.DoIP-071 AL – DoIP entity diagnostic message NACK code set to 03_{16}
Each DoIP entity shall send the diagnostic message negative acknowledgement with NACK code set to 03_{16} (see Table 26) when the diagnostic message contains an unknown target address (e.g. server DoIP entity not connected to the addressed DoIP gateway).	

REQ	7.DoIP-072 AL – DoIP entity diagnostic message NACK code set to 04_{16}
Each DoIP entity shall send the diagnostic message negative acknowledgement with NACK code set to 04_{16} (see Table 26) when the diagnostic message exceeds the maximum supported length of the transport protocol of the target network or target server DoIP entity (e.g. messages larger than 4095 bytes on CAN or when an server DoIP entity-specific message size limit is exceeded).	

NOTE 1 This implies that a NACK is also sent if a functionally addressed DoIP message has to be routed to different subnetworks (e.g. TA is set to functional group address $E000_{16}$) and one or more subnetworks do not support the diagnostic message payload length.

EXAMPLE If a functionally addressed DoIP message is routed through several sub-networks including a CAN sub-network and the DoIP diagnostic message payload size exceeds 7 bytes, the limitation for functionally addressed requests on CAN applies (only single frames). The DoIP gateway sends a NACK with NACK code = 04_{16} and discards the DoIP diagnostic request message.

REQ	7.DoIP-073 AL – DoIP entity diagnostic message NACK code set to 05_{16}
Each DoIP entity shall send the diagnostic message negative acknowledgement with NACK code set to 05_{16} (see Table 26) when the diagnostic message is too large to be copied into the destination buffer (e.g. the transport protocol refuses the request to provide the necessary buffer).	

NOTE 2 This can be a temporary problem if a DoIP gateway uses dynamic buffer allocation.

Table 26 — Diagnostic message negative acknowledge codes

Value	Description	Support
00_{16} to 01_{16}	Reserved by this document	—
02_{16}	Invalid source address	mandatory
03_{16}	Unknown target address	mandatory
04_{16}	Diagnostic message too large	mandatory
05_{16}	Out of memory	mandatory
06_{16}	Target unreachable	optional
07_{16}	Unknown network	optional
08_{16}	Transport protocol error	optional
09_{16} to FF_{16}	Reserved by this document	—

REQ	7.DoIP-103 AL – DoIP entity diagnostic message NACK code set to 06_{16}
If supported, each DoIP entity shall send the diagnostic message negative acknowledgement with NACK code set to 06_{16} (see Table 26) when the target address points to a device that cannot currently be reached.	

NOTE 3 This can be due to an unavailable destination network (e.g. temporary network reorganization or physical fault).

REQ	7.DoIP-107 AL – DoIP entity diagnostic message NACK code set to 07_{16} or 08_{16}
If supported and if an unknown target network or transport protocol error occurs that is not covered by the previous NACK codes, the DoIP entity shall send the diagnostic message negative acknowledgement with NACK code set to 07_{16} or 08_{16} (see Table 26).	

REQ	7.DoIP-074 AL – DoIP entity discard the received diagnostic message
Each DoIP entity shall discard the received diagnostic message if any of the aforementioned diagnostic message negative acknowledgement conditions (requirement DoIP-071 to DoIP-073, DoIP-103, DoIP-107) applies.	

9.6 AL alive check request and alive check response

This subclause specifies the message structures of the DoIP messages that are used to determine whether an open TCP_DATA socket is still in use by client DoIP entity. The alive check messages are utilized by the TCP_DATA socket handler (see 12.6.4). Figure 10 shows an example sequence of client DoIP entity triggering alive check messages while trying to establish a new TCP_DATA socket.

REQ	7.DoIP-075 AL – DoIP entity alive check request structure
Each DoIP entity shall support the alive check request as specified in Table 27.	

Table 27 — Payload type alive check request structure

Item	Pos.	Len.	Description	Values	Support
No additional message element					

REQ	7.DoIP-076 AL – DoIP entity alive check request message
Each DoIP entity shall send an alive check request according to the requirements in 12.6.4.	

REQ	7.DoIP-077 AL – DoIP entity alive check response structure
Each DoIP entity shall support the alive check response as specified in Table 28.	

Table 28 — Payload type alive check response structure

Item	Pos.	Len.	Description	Values	Support
Source address (SA)	0	2	Contains the logical address of the client DoIP entity that is currently active on this TCP_DATA socket.	See Table 13.	mandatory

REQ	7.DoIP-078 AL – DoIP entity alive check response message
Each DoIP entity shall receive and process alive check response messages according to the requirements in 12.6.4.	

NOTE The alive check response message can also be used by the client DoIP entity to keep a currently idle connection alive, i.e. it can be sent by the client DoIP entity even if it has not previously received an alive check request from a DoIP entity.

10 Transport layer security (TLS)

10.1 TLS secure diagnostic communication

TLS allows to establish an authenticated (ensures authenticity and integrity) and encrypted (confidentiality protection) communication channel between the client DoIP entity and the server DoIP entity.

If only the server DoIP entity performs a TLS authentication against the client DoIP entity and not vice versa in addition, then the application of the server DoIP entity saves additional calculation and resources. Verification of the client DoIP entity authenticity may be achieved in the diagnostic application layer and is left to the VM's discretion (e.g. usage of the UDS authentication service 29_{16} as specified in ISO 14229-1).

The secure diagnostic communication implements the TLS protocol for the transport channel between client DoIP entity and a server DoIP entity.

Before the client DoIP entity and the server DoIP entity begin exchanging DoIP messages over TLS, first the secured TCP connection is negotiated with the TLS handshake: the client DoIP entity and the server DoIP entity agree on the version of the TLS protocol, choose the common cipher suite, verify certificates, and complete the TLS session key exchange.

Figure 18 shows a simplified TLS handshake procedure.

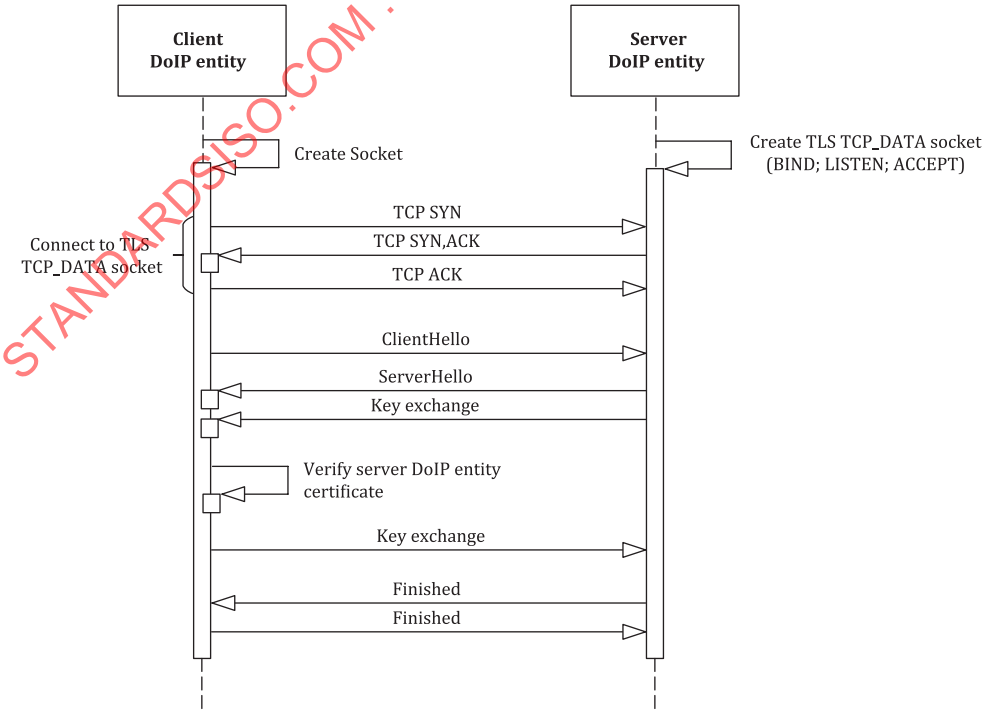


Figure 18 — TLS handshake (simplified)

REQ	4.DoIP-163 TL – Client DoIP entity is TLS client
The client DoIP entity shall act as a TLS client DoIP entity when initiating the TLS handshake.	

REQ	4.DoIP-164 TL – Server DoIP entity is TLS server
The server DoIP entity shall act as a TLS server when performing the TLS handshake.	

REQ	4.DoIP-165 TL – Server DoIP entity supports one TLS version
Each DoIP entity shall at least support one TLS version specified in the TLS DoIP application profile (see 10.2).	

REQ	4.DoIP-166 TL – Client DoIP entity supports several TLS versions
The client DoIP entity shall support several TLS versions specified in the TLS DoIP application profile (see 10.2).	

NOTE 1 The TLS DoIP application profile (see 10.2) defines the list of accepted TLS versions and the prioritized list of the supported TLS cipher suites for DoIP entities and the client DoIP entity.

NOTE 2 The TLS version negotiation is designed to be future-compatible. According to the TLS requirements, the client DoIP entity starts with the highest supporting TLS version. The server DoIP entity is still able to reply with whatever version the DoIP entity supports. For example, if a client DoIP entity sends a higher TLS version to a TLS 1.2 capable server DoIP entity, the handshake continues based on TLS 1.2.

The latest TLS 1.3 version provides the highest level of security.

REQ	4.DoIP-167 TL – Server DoIP entity support of at least one cipher suite
Depending on the utilized TLS version, the server DoIP entity shall support at least one cipher suite for the related TLS version listed in the referenced TLS DoIP application profile (see 10.2).	

REQ	4.DoIP-168 TL – Client DoIP entity support of several cipher suites
Depending on the utilized TLS version, the client DoIP entity shall support the cipher suites for the related TLS version listed in the referenced TLS DoIP application profile (see 10.2).	

Depending on the TLS authentication type, the client DoIP entity and server DoIP entity may require the security information as specified in Table 29.

Table 29 — TLS authentication type

TLS authentication	Requirements to client DoIP entity	Requirements to DoIP entity
Server authentication	At least the availability of the certificate of the CA which issued the DoIP entity certificate to check the authenticity of the DoIP entity.	At least the DoIP entity certificate with the corresponding private key. Further Intermediate certificates can be sent to the client DoIP entity in order to verify the complete chain of certificates.
Client authentication (optional for DoIP)	Additionally to server DoIP entity authentication requirements, the client DoIP entity requires the personal certificate issued by the intermediate CA and the corresponding private key.	Additionally to server DoIP entity authentication requirements, the DoIP entity requires the intermediate certificate of the CA which issued the client DoIP entity certificate.

REQ	4.DoIP-169 TL – Client DoIP entity supports server authentication
The client DoIP entity shall execute server authentication according to the DoIP server's implemented TLS version, which is supported by each server DoIP entity.	

REQ	4.DoIP-170 TL – Server DoIP entity certificate
Each server DoIP entity shall require at least the own DoIP entity certificate with the corresponding private key.	

REQ	4.DoIP-171 TL – Client DoIP entity certificate
If the client DoIP entity verifies the server DoIP entity's certificate during the TLS handshake then the client DoIP entity shall use the corresponding certificate chain, which includes the root certificate and intermediate certificates (same chain of trust) and the server DoIP entity certificate.	

REQ	4.DoIP-172 TL – Server DoIP entity TLS handshake
Within the TLS handshake, the server DoIP entity shall send the DoIP entity certificate. The root certificate shall not be transmitted.	

The certificate chain with the subordinate (related intermediate) certificate(s) can be transmitted.

REQ	4.DoIP-173 TL – Client DoIP entity TLS handshake
The client DoIP entity shall authenticate the server DoIP entity by verifying the DoIP entity certificate and the certificate chain of trust if provided within the TLS handshake.	

10.2 TLS DoIP application profile

10.2.1 TLS general

The TLS application profile defines the list of the accepted TLS versions and the prioritized list of the supported TLS cipher suites for DoIP entities and the client DoIP entity.

10.2.2 TLS accepted TLS versions for DoIP

REQ	4.DoIP-175 TL – Accepted TLS versions
Accepted TLS versions for DoIP shall be: — TLS 1.2 as specified in IETF RFC 5246 — TLS 1.3 as specified in IETF RFC 8446	

10.2.3 TLS accepted cipher suites

REQ	4.DoIP-176 TL – Accepted TLS 1.2 cipher suites
For TLS 1.2 version the cipher suites specified in Table 30 shall be accepted.	

Table 30 — TLS 1.2 version cipher suites

Cipher suite	RFC
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	IETF RFC 5289
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	IETF RFC 5289
TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256	IETF RFC 7905
TLS_ECDHE_ECDSA_WITH_AES_128_CCM	IETF RFC 7251
TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8	IETF RFC 7251
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256	IETF RFC 4492
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA384	IETF RFC 5289
TLS_ECDHE_ECDSA_WITH_NULL_SHA	IETF RFC 4492

The authentication-only cipher suite (with NULL encryption algorithm) does provide authenticity and integrity checks but not confidentiality for the TLS traffic. This can be practical for testing/debugging and authentication-only communication. It is recommended to disable authentication-only cipher suites in operational communication. In TLS 1.3, authentication-only cipher suites are not allowed anymore.

REQ	4.DoIP-177 TL – Accepted TLS 1.3 cipher suites
For TLS 1.3 version the cipher suites specified in Table 31 shall be accepted.	

Table 31 — TLS 1.3 version cipher suites

Cipher suite	RFC
TLS_AES_128_GCM_SHA256	IETF RFC 8446
TLS_AES_256_GCM_SHA384	IETF RFC 8446
TLS_CHACHA20_POLY1305_SHA256	IETF RFC 8446
TLS_AES_128_CCM_SHA256	IETF RFC 8446
TLS_AES_128_CCM_8_SHA256	IETF RFC 8446

10.2.4 TLS accepted TLS extensions

REQ	4.DoIP-178 TL – Supported TLS 1.2 extensions
Table 32 specifies TLS 1.2 version extensions, which shall be supported.	

Table 32 — TLS 1.2 version supported TLS extensions

TLS extension	RFC
Record_size_limit	IETF RFC 8449

Table 33 specifies the TLS 1.2 version extensions, which are optional.

Table 33 — TLS 1.2 version optional TLS extensions

TLS extension	RFC
Encrypt_then_MAC: Mandatory for ciphersuites using CBC mode. Not applicable for AEAD.	IETF RFC 7366
server_name: Server name with length 0 is allowed.	IETF RFC 6066

REQ	4.DoIP-179 TL – Not supported TLS 1.2 extensions
Table 34 specifies TLS 1.2 version extensions, which shall not be supported.	

Table 34 — TLS 1.2 version not supported TLS extensions

TLS extension	RFC
Client certificate URL	IETF RFC 6066
trusted_ca_keys	IETF RFC 6066
truncated_hmac	IETF RFC 6066
Renegotiation indication	IETF RFC 5746
Session hash and extended master secret extension	IETF RFC 7627
ClientHello padding extension	IETF RFC 7685
Data compression	IETF RFC 5246
Signature algorithms	IETF RFC 5246

REQ	4.DoIP-180 TL – Supported TLS 1.3 extensions
Table 35 specifies TLS 1.3 version extensions, which shall be supported (see IETF RFC 8446:2018, 9.2).	

Table 35 — TLS 1.3 version supported TLS extensions

TLS extension	RFC
Record_size_limit	IETF RFC 8449
Supported_versions (only TLS 1.2)	IETF RFC 8446

Table 36 specifies the TLS 1.3 version extensions, which are optional.

Table 36 — TLS 1.3 version optional TLS extensions

TLS extension	RFC
Server_name_identification: Server name with length 0 is allowed.	IETF RFC 6066

REQ	4.DoIP-181 TL – Not supported TLS 1.3 extensions
Table 37 specifies the TLS 1.3 version extensions, which shall not be supported.	

Table 37 — TLS 1.3 version not supported TLS extensions

TLS extension	RFC
0-RTT	IETF RFC 8446
Cookies	IETF RFC 8446

11 Transport layer (TL)

11.1 TL transmission control protocol (TCP)

The transmission control protocol (TCP) is a connection-oriented protocol, where applications on networked hosts can establish connections to one another to exchange data. The protocol guarantees

reliable and in-order delivery of sender-to-receiver data. Additionally, TCP provides flow control and congestion control. This document does not specify the specific algorithm that should be used. The TCP based communication between the two peers (client DoIP entity and the server DoIP entity) may be secured using an authenticated and optionally encrypted TLS connection. TCP and TLS are located on the transport layer, in accordance with the OSI layered architecture model (see Table 38).

Table 38 — TCP on OSI layers

OSI layer	Protocol
Transport	TCP, TLS
Network	IP (IPv4, IPv6)
Data link/Physical	e.g. Ethernet (ISO/IEC/IEEE 8802-3)

REQ	4.DoIP-114 TL – IPv4 and IPv6 TCP IETF RFC 793
Each IPv4 and IPv6 DoIP entity shall implement TCP as specified in IETF RFC 793.	

REQ	4.DoIP-115 TL – IPv4 and IPv6 TCP IETF RFC 1122
Each IPv4 and IPv6 DoIP entity shall implement the TCP-related requirements specified in IETF RFC 1122.	

REQ	4.DoIP-145 TL – IPv6 TCP retransmission timer computation
Each IPv6 DoIP entity shall implement the TCP retransmission timer computation defined in IETF RFC 6298.	

REQ	4.DoIP-157 TL – DoIP entity IPv4 and IPv6 TCP TLS
Each IPv4 and IPv6 DoIP entity that supports secured TCP communication shall implement TLS as specified in Clause 10.	

REQ	4.DoIP-162 TL – Client DoIP entity IPv4 and IPv6 TCP TLS
In order to setup a DoIP session with a DoIP entity that supports secured TCP communication, the client DoIP entity shall implement TLS as specified in Clause 10.	

TCP uses a pair of port numbers to identify a connection. The remote port is the port on the communication partner you send and receive messages from. The local port is the port in the machine you send messages from and receive nessages on. The ports listed in Table 39 are the receiving ports on the DoIP entities, that are used for TCP connections between client DoIP entity and server DoIP entities. The port numbers of the DoIP entity are different for unsecured and secured communication.

Table 39 — Supported TCP ports

Name	Protocol	Port number	Description	Support condition
TCP_DATA	TCP (unicast)	13400 for unsecured communication 3496 for secured (TLS) communication (see [10] for further information)	DoIP routing messages from the client DoIP entity to the vehicle servers (e.g. diagnostic requests) and vice versa (e.g. diagnostic responses)	mandatory

REQ	4.DoIP-001 TL – DoIP entity listens to port TCP_DATA
For unsecured TCP communication, each DoIP entity shall listen to port TCP_DATA as specified in Table 39 in order to establish communication with a client DoIP entity trying to connect on the TCP port.	

REQ	4.DoIP-002 TL – DoIP entity supports TCP data sockets
For unsecured TCP communication, each DoIP entity shall support <n+1> TCP data sockets, where <n> is the number of concurrent TCP data connections supported by the respective DoIP entity.	

REQ	4.DoIP-158 TL – DoIP entity listens on TLS port TCP_DATA
For secured TCP communication, each DoIP entity shall listen on TLS port TCP_DATA as specified within Table 39 in order to establish a secured communication with the client DoIP entity trying to connect on the TLS port.	

REQ	4.DoIP-159 TL – DoIP entity supports TLS data sockets
For secured TCP communication, each DoIP entity shall support <k+1> TLS data sockets, where <k> is the number of concurrent TLS data connections supported by the respective DoIP entity.	

REQ	4.DoIP-003 TL – Client DoIP entity TCP data sockets
For unsecured TCP communication, the client DoIP entity shall be capable of supporting <m> TCP data connections (TCP data sockets).	

The local port (i.e. source port) is chosen automatically during socket creation; the remote port is defined by the TCP_DATA port on the vehicle.

REQ	4.DoIP-160 TL – Client DoIP entity TLS data sockets
For secured TCP communication, the client DoIP entity shall be capable of supporting <j> TLS data connections (TLS data sockets).	

The TLS local port (i.e. source port) is chosen automatically during socket creation; the TLS remote port is defined by the TCP_DATA port on the vehicle.

NOTE A particular DoIP entity can support all combinations of unsecured and secured TCP communication: only unsecured TCP communication, only secured TCP communication or both unsecured and secured TCP communication (e.g. if the DoIP entity is emission related and supports VOBD).

Figure 19 shows the TCP socket states.

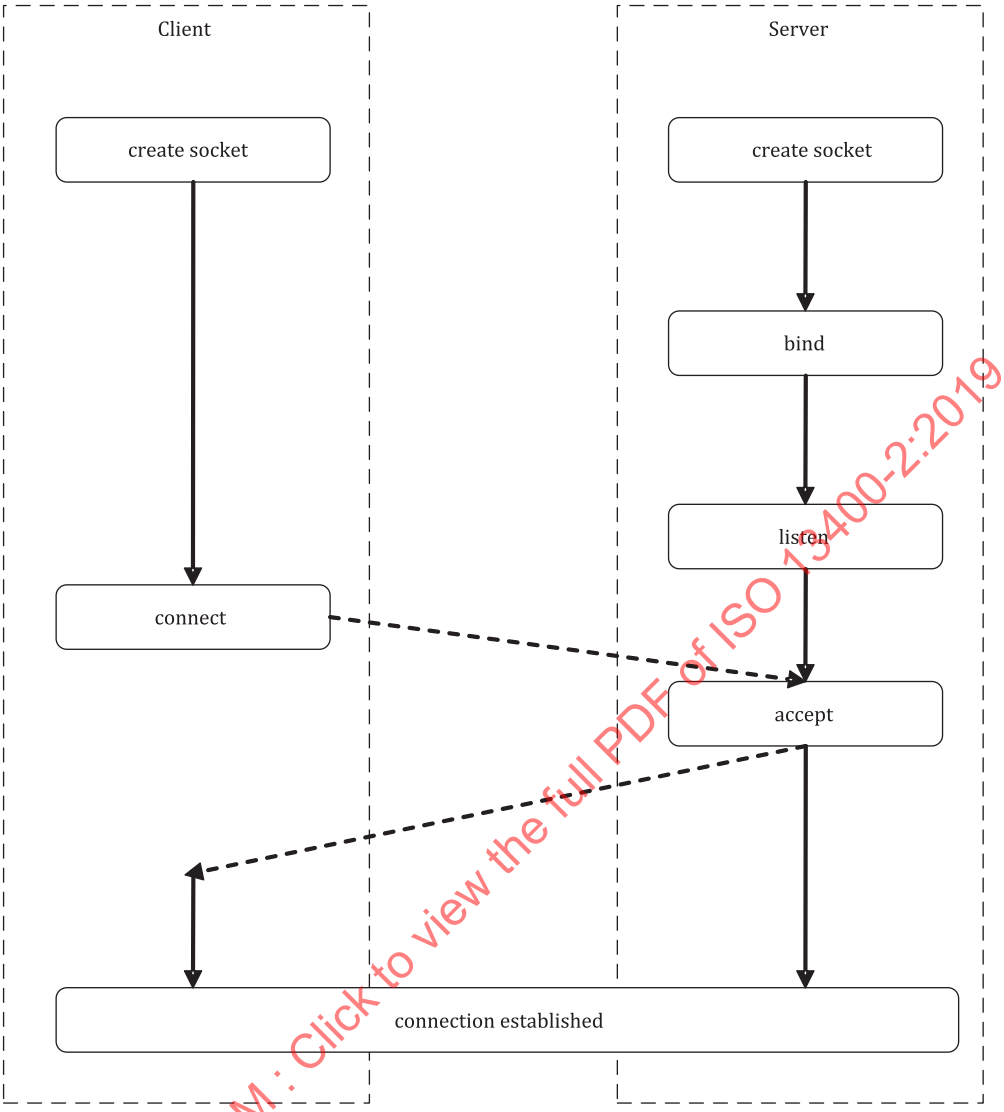


Figure 19 — TCP socket states

11.2 TL user datagram protocol (UDP)

The user datagram protocol (UDP) is a connectionless protocol. UDP does not provide the reliability and ordering guarantees that TCP does. Packets may arrive out of order or may be lost without notification of the sender or receiver. UDP is faster and more efficient for many lightweight or time-sensitive purposes. UDP is located on the transport layer of the OSI layered architecture model (see Table 40).

Table 40 — UDP on OSI layers

OSI layer	Protocol
Transport	UDP (IETF RFC 768)
Network	IP (IPv4, IPv6)
Data link/Physical	e.g. Ethernet (ISO/IEC/IEEE 8802-3)

REQ	4.DoIP-006 TL – DoIP entity UDP IETF RFC 768
Each DoIP entity shall implement UDP as specified in IETF RFC 768.	

REQ	4.DoIP-007 TL – DoIP entity UDP IETF RFC 1122
Each DoIP entity shall implement the UDP-related requirements in IETF RFC 1122.	

REQ	4.DoIP-161 TL – Unsecured UDP DoIP communication
UDP protocol based DoIP communication shall not be secured.	

UDP ports are used to identify the specific usage of UDP packets. The UDP ports specified in Table 41 are used for vehicle information services and control commands sent using UDP packets (e.g. when broadcasting requests to a local network).

Table 41 — UDP ports

Name	Protocol	Port number	Description	Support condition
UDP_DISCOVERY	UDP	13400 (see [10] for further information)	Used for vehicle information requests and control commands from the client DoIP entity to the vehicle's DoIP entities. This port is used as the destination port in UDP packets sent by the client DoIP entity. Used for UDP packets sent by the DoIP entities without having received a request (e.g. vehicle announcement message). This port is used as the destination port in these UDP packets. The source port for these UDP packets may be UDP_DISCOVERY but can also be assigned dynamically.	mandatory
UDP_TEST_EQUIPMENT_REQUEST	UDP	Dynamically assigned	This port is assigned dynamically by the client DoIP entity and used as the source port in UDP packets when transmitting messages (destination port set to UDP_DISCOVERY) to DoIP entities. This port is used as the destination port in UDP packets sent by the DoIP entities as response to the corresponding message. The source port for these UDP packets is set to UDP_DISCOVERY but can also be assigned dynamically.	mandatory

NOTE 1 Table 41 does not list the UDP ports, which are needed to implement other standard protocols specified in this document. Only the additional ports utilized by DoIP communication are specified.

REQ	4.DoIP-008 TL – Server DoIP entity listens on port UDP_DISCOVERY
Each server DoIP entity shall listen to port UDP_DISCOVERY as specified in Table 41.	

REQ	4.DoIP-009 TL – Server DoIP entity transmits on port UDP_DISCOVERY
Each server DoIP entity shall transmit UDP messages with the destination port set to UDP_DISCOVERY as specified in Table 41 in order to send unsolicited DoIP messages (e.g. vehicle announcement message).	

REQ	4.DoIP-010 TL – Client DoIP entity listens on port UDP_DISCOVERY
The client DoIP entity shall listen to port UDP_DISCOVERY as specified in Table 41 in order to be able to receive unsolicited DoIP messages.	

NOTE 2 As unsolicited messages are always transmitted to the one listening port at the client DoIP entity (i.e. UDP_DISCOVERY), some kind of middleware might be required to distribute the gathered information (e.g. vehicle announcement) to all interested applications that can be reached by the same IP address. Alternatively, the local port is used by multiple applications located on the client DoIP entity by using a reuse port option (e.g. SO_REUSEPORT) as long as only multicast messages are expected.

REQ	4.DoIP-011 TL – Client DoIP entity transmits on port UDP_DISCOVERY
The client DoIP entity shall transmit UDP messages to the DoIP entity with the UDP destination port set to UDP_DISCOVERY.	

REQ	4.DoIP-135 TL – Client DoIP entity transmits on port UDP_TEST_EQUIPMENT_REQUEST
The client DoIP entity shall transmit UDP messages to the DoIP entity with the UDP source port UDP_TEST_EQUIPMENT_REQUEST dynamically assigned within the dynamic port range (49 152 to 65 535).	

REQ	4.DoIP-136 TL – Client DoIP entity listens on port UDP_TEST_EQUIPMENT_REQUEST
The client DoIP entity shall listen to port UDP_TEST_EQUIPMENT_REQUEST specified in Table 41 for at least the time A_DoIP_Ctrl after the request has been transmitted in order to be able to receive responses to the previous UDP request messages. The port UDP_TEST_EQUIPMENT_REQUEST shall be in the listen state before sending a DoIP request message on this port to DoIP entities.	

REQ	4.DoIP-137 TL – Server DoIP entity transmits on port UDP_TEST_EQUIPMENT_REQUEST
Each server DoIP entity shall transmit UDP messages with the destination port set to UDP_TEST_EQUIPMENT_REQUEST as specified in Table 41 in order to respond to messages, which were received through port UDP_DISCOVERY.	

Depending on the implementation of the client DoIP entity, either the dynamically assigned UDP_TEST_EQUIPMENT_REQUEST port is assigned once during or before the first transmission of a UDP packet to a server DoIP entity or it is dynamically re-assigned for each individual UDP request message and response. Also, depending on whether messages are sent repeatedly, response messages might arrive asynchronously and might no longer be associated with the specific corresponding request. In this case, it is up to the application of the client DoIP entity to ensure that it can handle these situations (e.g. keep transmitting vehicle identification request messages until the first vehicle identification response message arrives and then ignore the remaining arriving vehicle identification responses). In the case of multiple client DoIP entity instances behind one IP address, it is recommended that the different applications select different UDP_TEST_EQUIPMENT_REQUEST port numbers, in order to simplify mapping the response message to the matching request message.

Figure 20 depicts the UDP port usage for unsolicited DoIP messages.

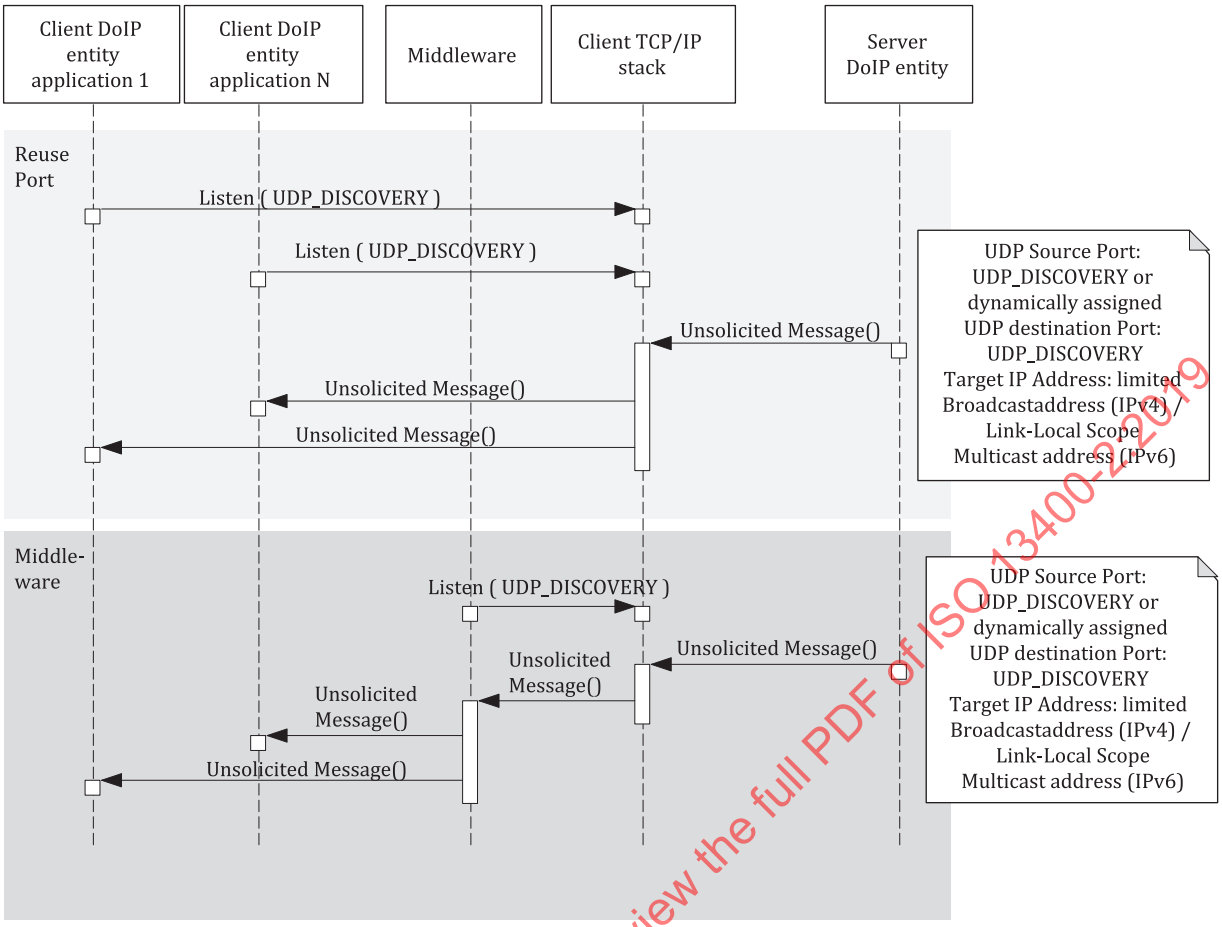


Figure 20 — UDP port usage for unsolicited DoIP messages

Figure 21 depicts the UDP port usage for DoIP request and response messages.

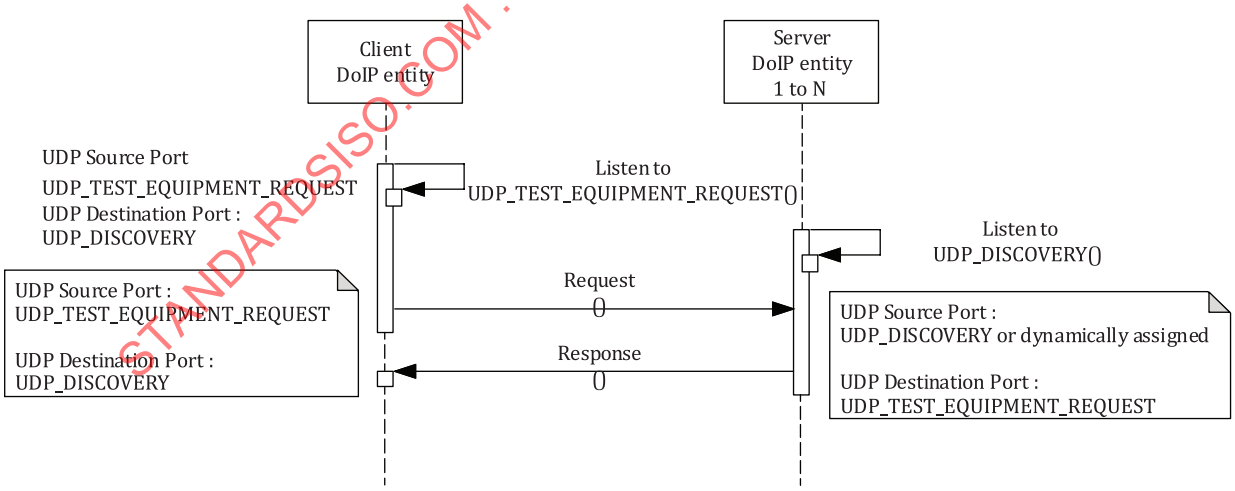


Figure 21 — UDP port usage for DoIP request and response messages

11.3 TL handling of UDP messages

The requirement specifies the handling of UDP messages.

REQ	4.DoIP-122 TL – DoIP entity transmits one DoIP message per UDP datagram
A DoIP entity shall only transmit one DoIP message per UDP datagram.	

12 Network layer (NL)

12.1 NL internet protocol (IP)

The protocol specified in this document is based on the Internet protocol (IP) standards known as IPv4 (see IETF RFC 791) and IPv6 (see IETF RFC 2460). Although the mandatory features of this document are intended to be based on IPv6 only, use of IPv4 is specified for applications of this communication protocol in network areas where backward compatibility to IPv4 is required. The IP is datagram based, unreliable and located on the network layer in accordance with the OSI layered architecture model (see Figure 1).

The process of how a node acquires an IP address is specified in 9.1.2. Table 42 defines the IPv4/IPv6 on OSI layers.

Table 42 — IPv4/IPv6 on OSI layers

OSI layer	Protocol	
Network	IPv6 (IETF RFC 2460; preferred)	IPv4 (IETF RFC 791; for backward compatibility reasons only)
Data link/Physical	e.g. Ethernet (ISO/IEC/IEEE 8802-3)	

REQ	3.DoIP-109 NL – Same IP version on vehicle wireline network
All DoIP entities on a vehicle wireline network shall implement the same Internet protocol version, either IPv4 in accordance with IETF RFC 791 or IPv6 in accordance with IETF RFC 2460.	

It is recommended that IPv6 is used in order to benefit from the advantages (e.g. link local IP address assignment; faster forwarding through routers) of this protocol version. IPv4 may only be used for backward compatibility reasons (e.g. for integration into existing dealership IP networks). The support of Jumbograms for IPv6 is optional and consequently compliance with IETF RFCs related to Jumbograms is not required in this document.

NOTE Interaction of the vehicle wireline DoIP entities with a future wireless IPv6 entity forms the subject of future International Standards.

12.2 NL IPv4 address resolution protocol (ARP)

The address resolution protocol (ARP) is a method for determining a host's hardware (MAC) address when only the host's IP address is known. They are also used to verify whether an IP address is in use by another host. ARP is located on the network layer, in accordance with the OSI layered architecture model (see Table 43).

Table 43 — ARP on OSI layers

OSI layer	Protocol
Network	IPv4: ARP (IETF RFC 826)

Table 43 *(continued)*

OSI layer	Protocol
Data link/Physical	e.g. Ethernet (ISO/IEC/IEEE 8802-3)

REQ	3.DoIP-110 NL – IPv4 ARP
If IPv4 is used, each DoIP entity shall implement ARP as defined in IETF RFC 826.	

Each host, that implements IPv4 also implements ARP, as it is an essential part of IPv4 communication over Ethernet-based networks. Implementation of the reverse address resolution protocol (RARP) is not required as this requires a RARP server as part of the network, which is not mandatory in IPv4 networks.

12.3 NL IPv6 neighbour discovery protocol (NDP)

The neighbour discovery protocol (NDP) is a method for determining a host's hardware (MAC) address when only the host's IP address is known. It is also used to verify whether an IP address is in use by another host. NDP is located on the network layer, in accordance with the OSI layered architecture model (see Table 10).

Table 44 — NDP on OSI layers

OSI layer	Protocol
Network	IPv6: NDP (IETF RFC 4861)
Data link/Physical	e.g. Ethernet (ISO/IEC/IEEE 8802-3)

REQ	3.DoIP-111 NL – IPv6 NDP
If IPv6 is used, each DoIP entity shall implement NDP as defined in IETF RFC 4861.	

12.4 NL internet control message protocol (ICMP)

The internet control message protocol (ICMP) is part of the IP suite and is used to send error messages, e.g. to indicate that a requested service is not available or that a host could not be reached. Consequently, ICMP is a mandatory part of an IP stack implementation and is located on the network layer, in accordance with the OSI layered architecture model (see Table 45).

Table 45 — ICMP on OSI layers

OSI layer	Protocol
Network	IPv4: ICMP (IETF RFC 792) IPv6: ICMP v6 (IETF RFC 4443)
Data link/Physical	e.g. Ethernet (ISO/IEC/IEEE 8802-3)

REQ	3.DoIP-112 NL – IPv4 ICMP
If IPv4 is used, each DoIP entity shall implement ICMP as specified in IETF RFC 792.	

REQ	3.DoIP-113 NL – IPv6 ICMPv6
If IPv6 is used, each DoIP entity shall implement ICMPv6 as specified in IETF RFC 4443.	