INTERNATIONAL STANDARD

ISO/IEC 18013-5

First edition
2021-09

Personal identification — ISO-compliant driving licence —

Part 5: Mobile driving licence (mDL) application

Identification des personnes — Permis de conduire conforme à l'ISO —

Partie 5: Application permis de conduire sur téléphone mobile

Reference number
ISO/IEC 18013-5:2021(E)

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

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

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 document 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 or www.iec.ch/members_experts/refdocs).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC 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) or the IEC list of patent declarations received (see https://patents.iec.ch).

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. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 17, *Cards and security devices for personal identification*.

A list of all parts in the ISO/IEC 18013 series can be found on the ISO and IEC websites.

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 and www.iec.ch/national-committees.

Introduction

The ISO/IEC 18013 series establishes guidelines for the design format and data content of an ISO-compliant driving licence (IDL) with regard to human-readable features (ISO/IEC 18013-1), ISO machine-readable technologies (ISO/IEC 18013-2), access control, authentication and integrity validation (ISO/IEC 18013-3), and associated test methods (ISO/IEC 18013-4). It creates a common basis for international use and mutual recognition of the IDL without impeding individual countries/states in applying their privacy rules and national/community/regional motor vehicle authorities in taking care of their specific needs.

This document describes interface and related requirements to facilitate ISO-compliant driving licence (IDL) functionality on a mobile device. The requirements are specifically intended to enable verifiers not affiliated with or associated with the issuing authority to gain access to and authenticate the information. In addition, the requirements allow the holder of the driving licence to decide what information to release to a verifier. Other characteristics include the ability to update information frequently, and to authenticate information at a high level of confidence.

A mobile document conforming to this document primarily conveys the driving privileges associated with a person. It does so by providing means to associate the person presenting the mobile document with the mobile document itself. However, the transaction and security mechanisms in this document have been designed to support other types of mobile documents, specifically including identification documents. Consequently the mechanisms in this document can be used for any type of mobile identification document, regardless of the additional attributes the mobile document may convey. The details of the data elements to be used by other mobile documents are left to the respective issuing authority and are not within the scope of this document.

Personal identification — ISO-compliant driving licence —

Part 5:
Mobile driving licence (mDL) application

1 Scope

This document establishes interface specifications for the implementation of a driving licence in association with a mobile device. This document specifies the interface between the mDL and mDL reader and the interface between the mDL reader and the issuing authority infrastructure. This document also enables parties other than the issuing authority (e.g. other issuing authorities, or mDL verifiers in other countries) to:

— use a machine to obtain the mDL data;

— tie the mDL to the mDL holder;

— authenticate the origin of the mDL data;

— verify the integrity of the mDL data.

The following items are out of scope for this document:

— how mDL holder consent to share data is obtained;

— requirements on storage of mDL data and mDL private keys.

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 3166-1, *Codes for the representation of names of countries and their subdivisions — Part 1: Country code*

ISO 3166-2:2020, *Codes for the representation of names of countries and their subdivisions — Part 2: Country subdivision code*

ISO/IEC 5218, *Information technology — Codes for the representation of human sexes*

ISO/IEC 7816-4:2020, *Identification cards — Integrated circuit cards — Part 4: Organization, security and commands for interchange*

ISO/IEC 8859-1, *Information technology — 8-bit single-byte coded graphic character sets — Part 1: Latin alphabet No. 1*

ISO/IEC 18004, *Information technology — Automatic identification and data capture techniques — QR Code bar code symbology specification*

ISO/IEC 18013-1:2018, *Information technology — Personal identification — ISO-compliant driving licence — Part 1: Physical characteristics and basic data set*

ISO/IEC 18013-2:2020, *Personal identification — ISO-compliant driving licence — Part 2: Machine-readable technologies*

ISO/IEC 19785-3:2020, *Information technology — Common Biometric Exchange Formats Framework — Part 3: Patron format specifications*

BSI TR-03111, *Elliptic Curve Cryptography, Version 2.10, June 2018*

FIPS 186-4:2013, *D igital Signature Standard (DSS)*

NFC Forum, *Connection Handover (CH) Technical Specification, Version 1.5*

NIST SP 800-38D, *M. Dworkin, Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC, November 2007*

OpenID Foundation *OpenID Connect Core 1.0 incorporating errata set 1*

OpenID Foundation *OpenID Connect Discovery 1.0 incorporating errata set 1*

RFC 4122, *P. Leach et al., A Universally Unique IDentifier (UUID) URN Namespace, July 2005*

RFC 4648, *S. Josefsson, The Base16, Base32, and Base64 Data Encodings, October 2006*

RFC 5246, *T. Dierks et al., The Transport Layer Security (TLS) Protocol Version 1.2, August 2008*

RFC 5280, *D. Cooper et al., Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, May 2008*

RFC 5639, *M. Lochter et al., Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation, March 2010*

RFC 5869, *H. Krawczyk, HMAC-based Extract-and-Expand Key Derivation Function (HKDF), May 2010*

RFC 6066, *D. Eastlake 3rd, Transport Layer Security (TLS) Extensions: Extension Definitions, January 2011*

RFC 7049, *C. Bormann et al., Concise Binary Object Representation (CBOR), October 2013*

RFC 7515, *J. Bradley et al., JSON Web Signature (JWS), May 2015*

RFC 7518, *M. Jones et al., JSON Web Algorithms (JWA), May 2015*

RFC 7519, *J. Bradley et al., JSON Web Token (JWT), May 2015*

RFC 7748, *A. Langley et al., Elliptic Curves for Security, January 2016*

RFC 7905, *A. Langley et al., ChaCha20-Poly1305 Cipher Suites for Transport Layer Security (TLS), June 2016*

RFC 8032, *S. Josefsson et al., Edwards-Curve Digital Signature Algorithm (EdDSA), January 2017*

RFC 8152, *J. Schaad, CBOR Object Signing and Encryption (COSE), July 2017*

RFC 8252, *W. Denniss et al., Oauth 2.0 for Native Apps, October 2017*

RFC 8259, *T. Bray, The JavaScript Object Notation (JSON) Data Interchange Format, December 2017*

RFC 8410, *S. Josefsson et al., Algorithm Identifiers for Ed25519, Ed448, X25519, and X448 for Use in the Internet X.509 Public Key Infrastructure, August 2018*

RFC 8422, *Y. Nir et al., Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS) Versions 1.2 and Earlier, August 2018*

RFC 8943, *M. Jones et al., Concise Binary Object Representation (CBOR) Tags for Date, November 2020*

RFC, *CBOR Object Signing and Encryption (COSE): Headers for carrying and referencing X.509 certificates*

Wi-Fi Alliance, *Neighbor Awareness Networking Specification, Version 3.1*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

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

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

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

3.1
mobile device
portable computing device that at least:

— has a small form factor such that it can easily be carried by a single individual;

— is designed to operate, transmit and receive information without a wired connection;

— possesses local, nonremovable or removable data storage;

— includes a self-contained power source;

— includes a display;

— includes a means for the holder of the portable computing device to interact with the device

Note 1 to entry: Adapted from NIST SP 800-157.

3.2
mdoc
document or application that resides on a *mobile device* (3.1) or requires a mobile device as part of the process to gain access to the document or application

3.3
mdoc reader
device that can retrieve *mdoc* (3.2) data for verification purposes

3.4
mdoc holder
individual to whom an *mdoc* (3.2) is issued

3.5
mdoc verifier
person or organization using and/or controlling an *mdoc reader* (3.3) to verify an *mdoc* (3.2)

3.6
mDL
driving licence that fulfils at least the same function as an IDL but, instead of being paper or plastic based, is an *mdoc* (3.2)

Note 1 to entry: ISO-compliant driving licence (IDL) is defined in ISO/IEC 18013-1.

3.7
mDL reader
mdoc reader (3.3) that can retrieve *mDL* (3.6) data

3.8
mDL holder
individual to whom an *mDL* (3.6) is issued, i.e. legitimate holder of the driving privileges reflected on an mDL

3.9
mDL verifier
person or organization using and/or controlling an *mDL reader* (3.7) to verify an *mDL* (3.6)

3.10
licensing authority
authorized agent organisation that issues a driving licence

EXAMPLE National, federal, state, provincial, regional, territorial, or local Ministry of Transport, Department of Motor Vehicles, or Police Agency.

[SOURCE: ISO/IEC 18013-1:2018, 3.15]

3.11
issuing country
country which issued the driving licence or within which the *licensing authority* (3.10) is located

[SOURCE: ISO/IEC 18013-1:2018, 3.12, modified — The words "according to Annex F" have been removed.]

3.12
issuing authority
licensing authority (3.10), or *issuing country* (3.11) if separate licensing authorities have not been authorized

[SOURCE: ISO/IEC 18013-1:2018, 3.11]

3.13
issuing authority infrastructure
infrastructure under control of the *issuing authority* (3.12)

3.14
issuing authority CA
certificate authority operated by or on behalf of an *issuing authority* (3.12)

3.15
device retrieval
method of data retrieval exclusively using the interface between the *mdoc* (3.2) and the *mdoc reader* (3.3)

3.16
server retrieval
method of data retrieval using the interface between the *mdoc reader* (3.3) and the *issuing authority infrastructure* (3.13)

3.17
server retrieval token
token identifying the *mdoc holder* (3.4) and the *mdoc* (3.2) to the *issuing authority* (3.12)

3.18
PCD mode
mode in which an NFC-enabled *mobile device* (3.1) operates as a PCD

[SOURCE: ISO/IEC 14443-3:2018, 3.7, modified — The words "a PXD" have been replaced with "an NFC-enabled mobile device".]

3.19
PICC mode
mode in which an NFC-enabled *mobile device* (3.1) operates as a PICC

[SOURCE: ISO/IEC 14443-3:2018, 3.8, modified — The words "a PXD" have been replaced with "an NFC-enabled mobile device".]

4 Abbreviated terms

AES	advanced encryption standard
APDU	application protocol data unit
BLE	Bluetooth® low energy
BT SIG	Bluetooth special interest group
CA	certificate authority
CBOR	concise binary object representation
CDDL	concise data definition language
COSE	CBOR object signing and encryption
CSPRNG	cryptographically secure pseudo-random number generator
CRL	certificate revocation list
DER	distinguished encoding rules
DS	document signer
ECDH	elliptic curve Diffie-Hellman key agreement
ECDSA	elliptic curve digital signature algorithm
EdDSA	Edwards-curve digital signature algorithm
GATT	generic attribute profile
HKDF	HMAC-based extract-and-expand key derivation function
HMAC	hash-based MAC
IA	issuing authority
IACA	issuing authority certificate authority
IANA	internet assigned number authority
IDL	ISO-compliant driving licence
IKM	input keying material
JSON	JavaScript object notation
JWK	JSON web key
JWS	JSON web signature
JWT	JSON web token
KDF	key derivation function
MAC	message authentication code
MITM	man-in-the-middle attack

MSO	mobile security object
MTU	maximum transmission unit
NDEF	NFC data exchange format
NFC	near field communication
OCSP	online certificate status protocol
OID	object identifier
OIDC	OpenID connect
PCD	proximity coupling device
PICC	proximity card or object
PKI	public key infrastructure
RF	radiofrequency
RFU	reserved for future use
SHA	secure hash algorithm
TLS	transport layer security
URI	uniform resource identifier
URL	uniform resource locator
UTC	coordinated universal time
UUID	universally unique identifier
VICAL	verified issuer certificate authority list

5 Conformance requirement

An mDL is in conformance with this document if it meets all the requirements specified directly or by reference herein. Conformance with ISO/IEC 18013-1, ISO/IEC 18013-2, ISO/IEC 18013-3, and ISO/IEC 18013-4 is not required for conformance with this document, except for those clauses normatively referenced in this document.

An mDL reader is in conformance with this document if it meets all the requirements specified directly or referenced herein.

An issuing authority infrastructure is in conformance with this document if it meets all the requirements specified directly or referenced herein.

6 mDL overview

6.1 Interfaces

Figure 1 shows the interfaces in scope for this document. The explanation of each interface is as follows.

— Interface 1 in Figure 1 is the interface between the issuing authority infrastructure and the mDL. This interface is out of scope for this document.

— Interface 2 in Figure 1 is the interface between the mDL and the mDL reader. This interface is specified in this document. The interface can be used for connection setup and for the device retrieval method.

— Interface 3 in Figure 1 is the interface between the issuing authority infrastructure and the mDL reader. This interface is specified in this document. The interface can be used for the server retrieval method.

See Reference [9] for examples of use cases.

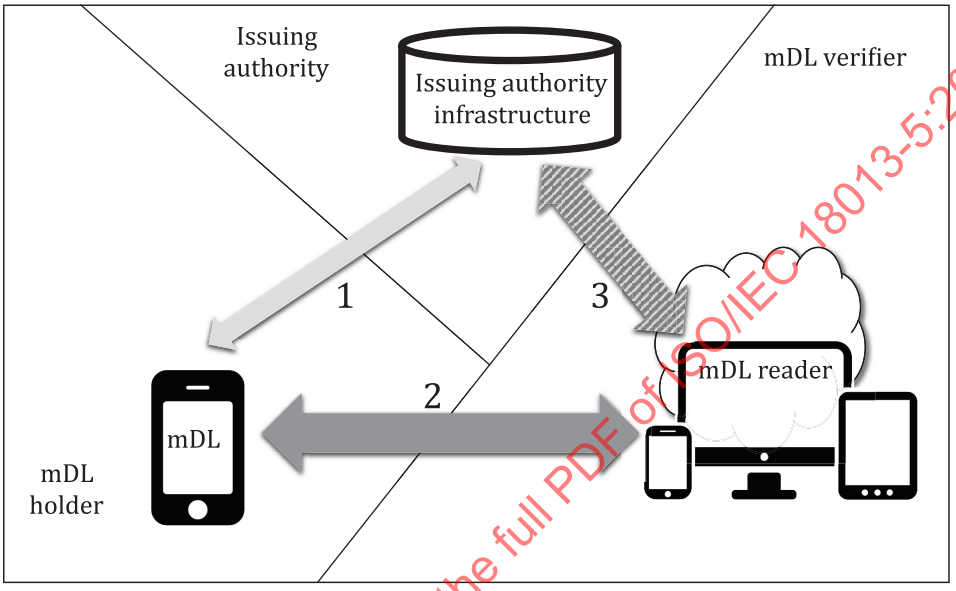

Figure 1 — mDL interfaces

6.2 Functional requirements

The functional requirements include at least the following.

a) An mDL verifier together with an mDL reader shall be able to request, receive and verify the integrity and authenticity of an mDL whether online connectivity is present or not for either the mDL or mDL reader.

b) An mDL verifier not associated with the issuing authority shall be able to verify the integrity and authenticity of an mDL.

c) An mDL verifier shall be enabled to confirm the binding between the person presenting the mDL and the mDL holder.

d) The interface between the mDL and the mDL reader shall support the selective release of mDL data to an mDL reader.

6.3 Technical requirements

6.3.1 Data model

The mDL data is organized as individual data elements which can be requested and returned independently from each other. The mDL data model is described in Clause 7. It describes the identifier and format of the data elements.

NOTE The concepts used in this document have been designed so that other mobile credentials, e.g. mobile identity or other credentials that substitute Table 5 with a different set of data elements, can also make use of the engagement and retrieval protocols described in this document. Specifically, the mdoc data model, which is illustrated in Figure 2, is based on elements with unique identifiers within a namespace. The number of elements can vary, and the model is indifferent to the value and data format of each element. As such the data model is generic and can apply to any kind of document.

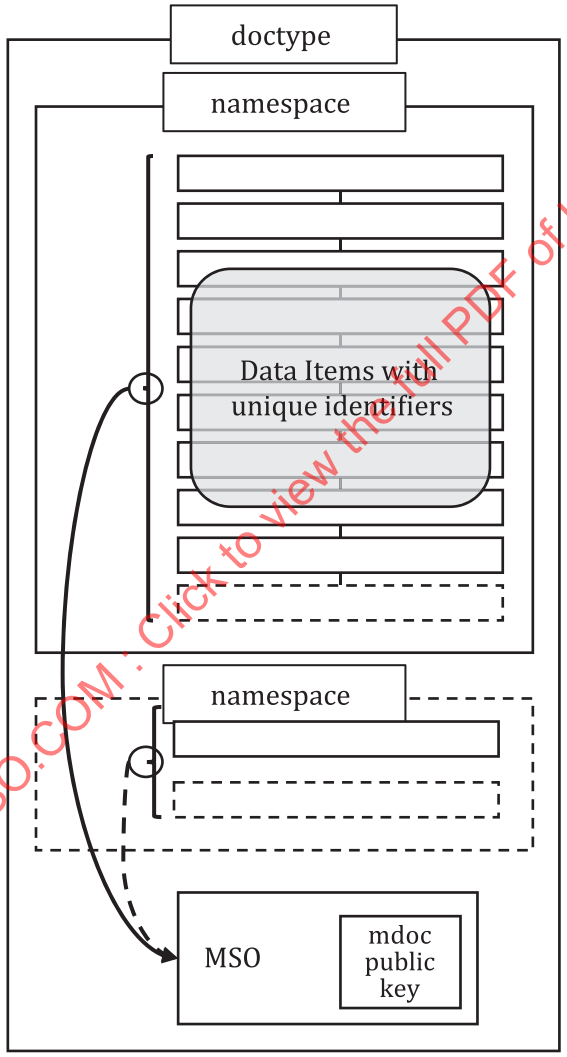

Figure 2 — mdoc data model

6.3.2 Data exchange

6.3.2.1 Overview

The mDL and mDL reader are implemented as an mdoc and mdoc reader, for which the requirements are described in Clause 8 and Clause 9.

Data exchange is divided into three phases: the initialization phase, the device engagement phase, and the data retrieval phase (as illustrated in Figure 3). After initialization between the mDL and the mDL reader three different transaction flows are distinguished:

— device engagement, followed by exchange of data using device retrieval between the mDL and the mDL reader [see (1) in Figure 3];

— device engagement, followed by exchange of server retrieval information using device retrieval between the mDL and the mDL reader, followed by exchange of data using server retrieval between the mDL reader and the issuing authority infrastructure [see (2) in Figure 3];

— device engagement, followed by exchange of data using server retrieval between the mDL reader and the issuing authority infrastructure [see (3) in Figure 3].

NOTE 1 For device retrieval, there is no requirement for any device involved in the transaction to be connected to the internet.

If the mDL reader receives the server retrieval token and URL from the mDL, either during device engagement or device retrieval, it may either use device retrieval or server retrieval. If it chooses to use device retrieval, either BLE, NFC or Wi-Fi Aware can be used to retrieve the information. If it chooses to use server retrieval, either OIDC or WebAPI can be used to retrieve the information.

NOTE 2 The transaction has been designed such that it is not necessary for the mDL holder to physically hand over the mobile device to the mDL verifier.

NOTE 3 The transaction protocols in this document provide generic means for a user to share connection information and optionally a server retrieval token.

The device data retrieval transport applies to any kind of data as it is designed to transport an encrypted blob.

The request and response commands (transported encrypted) are applicable to any kinds of document based on the mdoc data model and/or request for server retrieval token. Furthermore, the request and response commands are wallet compliant as elements from different documents can be requested and the response can include multiple documents from the same or different kinds.

The server retrieval method relies on OpenID Connect that is not specific to mDL, or on WebAPI that relies on the generic mdoc data model.

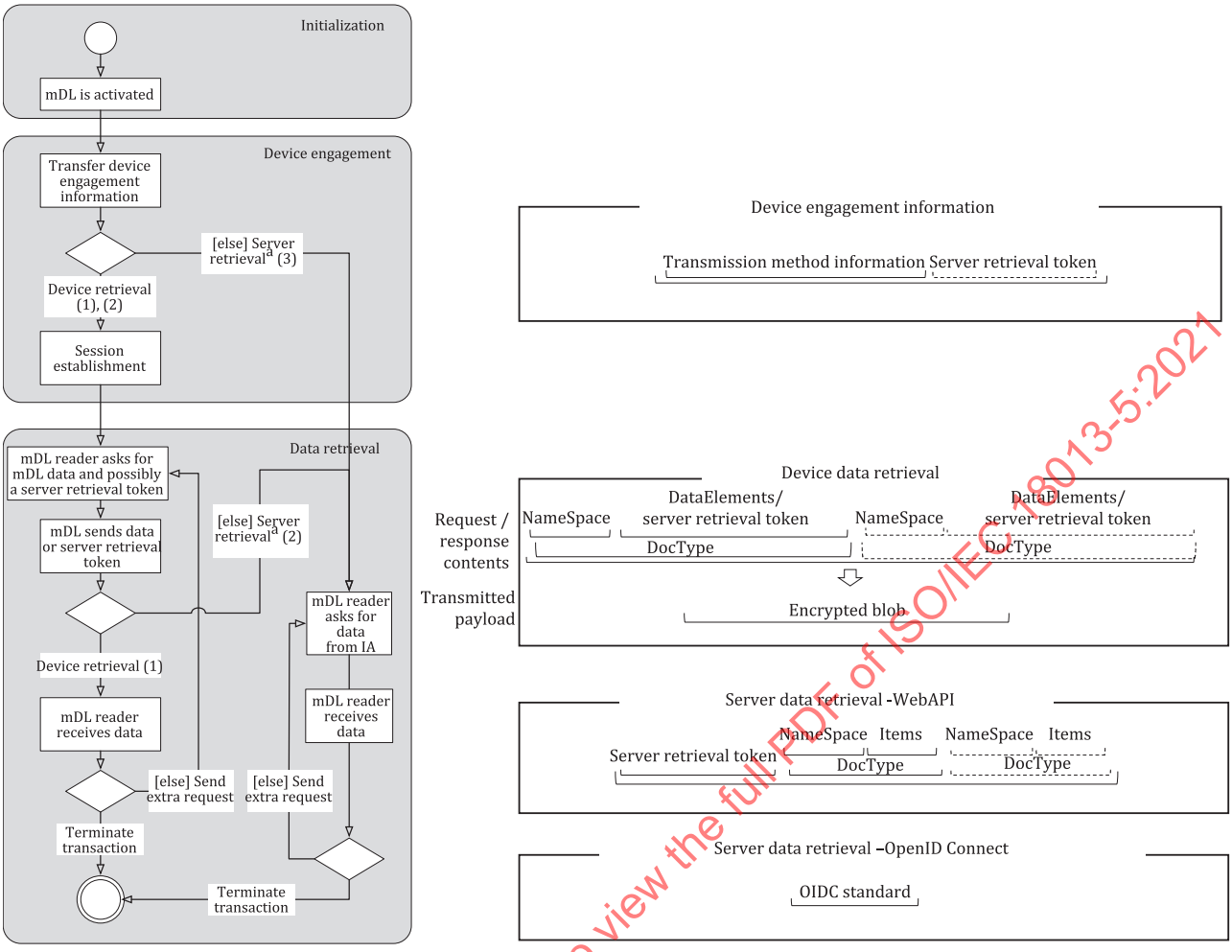

Figure 3 — mDL transaction flow

6.3.2.2 Initialization

During initialization, the mDL is activated. Activation is done by the mDL holder, or triggered by an mDL reader using NFC. Simultaneously, the mDL reader is activated. No requirements are specified for this phase.

NOTE It is important to avoid unauthorized access by an mDL reader if mDL activation is triggered by NFC.

6.3.2.3 Device engagement

During device engagement, information required to setup and secure data retrieval is exchanged between the mDL and the mDL reader. Transmission technologies available to transfer the device engagement data are as follows:

a) NFC,

b) QR code.

Table 1 shows the different transmission technologies for device engagement.

Table 1 — Device engagement technologies

Transmission technologies	Support		Reference
	mDL	mDL reader	
NFC	C[a]	M	8.2.2.1
QR code	C[a]	M	8.2.2.3

Key

C conditional

M mandatory

[a] Support for at least one of these methods is mandatory.

The device engagement information, described in 8.2.1, is transferred using one of the transmission technologies, described in 8.2.2.

To ensure that the device engagement is always possible, an mDL shall support at least one of the transmission technologies in Table 1. An mDL reader shall support all transmission technologies.

If the mDL supports NFC for device engagement, it shall support Static Handover, Negotiated Handover, or both, as described in 8.2.2.1. The mDL reader shall support both handover methods.

6.3.2.4 Data retrieval architecture

Figure 4 shows the different data retrieval interfaces and the flow of the messages.

When using device retrieval, the mDL and mDL reader communicate using mdoc request and mdoc response messages encoded with CBOR. These messages are transported using a data retrieval method. The data retrieval methods are agnostic to the information that is transferred.

After device engagement, if the mDL reader sets up a device retrieval connection, the mDL reader asks for data as defined in 8.3.2.1.2.1. The mDL sends an mdoc response according to 8.3.2.1.2.2. The mdoc request may include a request for server retrieval information used to perform server retrieval. If server retrieval information is requested next to other mDL data, the mDL shall return either the server retrieval information or the other requested data, but not both.

The different data retrieval methods are described in 6.3.2.5.

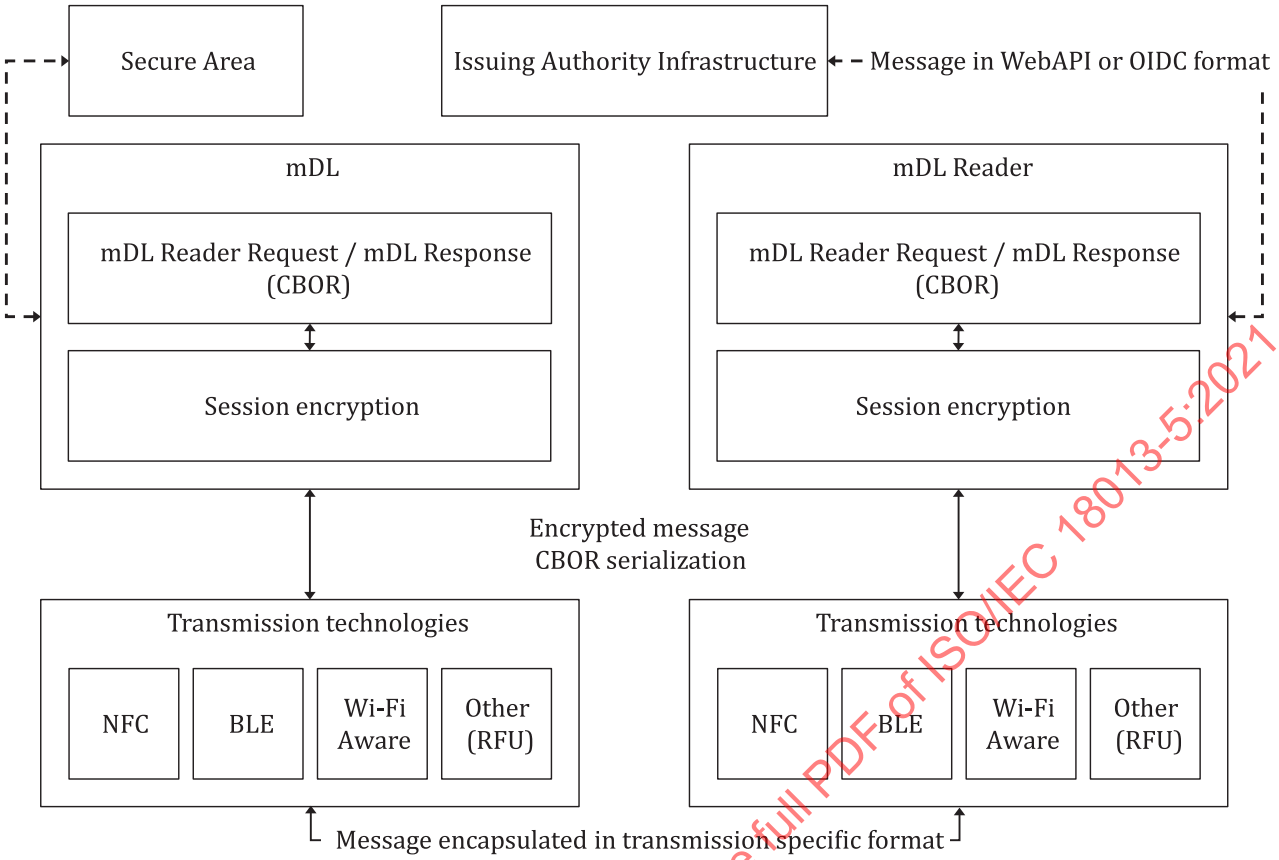

Figure 4 — Data retrieval architecture

NOTE 1 The secure area as present in Figure 4 indicates an area that provides additional protection of sensitive mDL related data. Security requirements regarding storage of credential information are outside the scope of this document. This includes the mDL private key and the mDL reader private key, if mdoc reader authentication or TLS client authentication is supported by the mDL reader. It is the responsibility of the issuing authority to ensure that all data stored on the mDL is stored securely.

NOTE 2 Implementation possibilities for a secure area are non-exhaustively listed in Clause E.5.

6.3.2.5 Data retrieval methods

The following methods are defined for retrieval of mDL data. Requirements for supporting these methods are defined in Table 2.

mDL data can be retrieved in two ways:

a) using device retrieval (interface 2 in Figure 1), see 8.3.2.1;

b) using server retrieval (interface 3 in Figure 1), see 8.3.2.2, where the server retrieval token may be retrieved by the mDL reader from the mDL during device engagement or during device retrieval.

Table 2 shows the transmission technologies and data retrieval methods.

Table 2 — Data retrieval methods

Data retrieval method	Transmission technology	Support			Reference
		mDL	mDL reader	issuing authority infrastructure	
Device retrieval	BLE	C [a]	M	N/A	8.3.3.1.1
	NFC	C [a]	M	N/A	8.3.3.1.2
	Wi-Fi Aware	O	R	N/A	8.3.3.1.3
Server retrieval	WebAPI	O	R	O	8.3.3.2.1
	OIDC	O	R	O	8.3.3.2.2

Key

M mandatory

C conditional

R recommended

O optional

N/A not applicable

[a] Support for at least one of these methods is mandatory.

To ensure that data retrieval is always possible, an mDL shall support device retrieval using BLE, NFC, or both transmission technologies. An mDL reader shall support the BLE and NFC transmission technology for device retrieval.

An mDL may support Wi-Fi Aware and an mDL reader should support Wi-Fi Aware.

An mDL and an issuing authority infrastructure may support WebAPI, OIDC or both and an mDL reader should support WebAPI and OIDC.

For device retrieval using BLE, the mDL reader shall support the mdoc central client mode and mdoc peripheral server mode, as defined in 8.3.3.1.1. The mDL and mDL reader may support the BLE L2CAP transmission profile as defined in Annex A.

All data retrieval methods shall use the data model as defined in Clause 7.

See Annex D for examples of data structures.

NOTE 1 If QR is used for device engagement and the mDL reader chooses to use NFC for data transfer, then there is no mechanism available for the mDL reader to indicate the choice for NFC data transfer to the mDL. It is possible that the mDL holder is not aware that the mDL needs to interface with the mDL reader using NFC. On the contrary, if NFC is used for device engagement, this problem does not exist.

NOTE 2 Due to the limited data transfer rate for NFC, if a large amount of data is required for a transaction, it is possible that it is neither practical nor reasonable to have an mDL holder hold the device within the RF range of the mDL reader for the duration of the transaction. Furthermore, due to the loss of signal when a device leaves the RF field, any mDL holder interactions with the mDL causing the mDL to leave the RF field require a new transaction to be initiated. This can be avoided by having all mDL holder interactions with the mDL done while the mDL stays in the field or if mDL does not require any mDL holder interactions while it is in the RF field.

6.3.3 Security mechanisms

The security of mDL data exchanged with an mDL reader is designed to preserve the triad of confidentiality, integrity, and authenticity by design and by default.

The security architecture aims to achieve four distinct goals.

a) Protection against forgery: data elements are signed by the issuing authority (IA). The degree of protection against forgery depends on the degree to which the IA's keys are protected. Minimizing the validity period of the data limits the value of the data.

b) Protection against cloning: the mDL produces a signature or message authentication code over session data. The private key used to authenticate the session data is stored only in the mDL. The corresponding public key in turn is signed by the IA. The degree of protection against cloning depends on the degree to which the mDL authentication key or the TLS server key is protected.

c) Protection against eavesdropping: communications between mDL and mDL reader are encrypted and authenticated. Device engagement uses a separate communication channel to mitigate the risk of Man in the Middle (MITM) attacks. In addition, the mDL reader can detect MITM attacks by validating the anti-cloning signature or message authentication code, which is created using a key for which the public part is signed by the IA in the mobile security object (MSO, see 9.1.2.4). If mdoc reader authentication is used, the mDL can detect MITM attacks before returning any data. Server retrieval uses TLS for encryption to further protect against eavesdropping and MITM attacks.

d) Protection against unauthorized access: an mDL is protected from unauthorized access by an mDL reader by multiple mechanisms. If device retrieval is used, the encryption key used for communications between the mDL and mDL reader is derived from an ephemeral key pair from both the mDL and mDL reader. The public key of the mDL is shared only through short-range device engagement. This ensures data is only transferred between the mDL and a particular mDL reader. The mDL can optionally authenticate the mDL reader by means of an mDL reader certificate and a signature created by the mDL reader using the corresponding private key. The mDL reader certificate is signed by a certificate authority trusted by the mDL for this purpose.

   Server retrieval uses a server retrieval token which can be used to ensure that data is only transferred between the IA infrastructure and a particular mDL reader. Moreover, the IA can optionally authenticate the mDL reader by means of TLS client authentication.

Revocation of an mDL is out of scope for this document. However, the MSO includes update information and validity time frames which enable the mDL reader to check the freshness of the data. The issuing authority shall define appropriate periods of validity that balance freshness with offline capability, taking into account that a shorter validity period mitigates certain security risks.

Table 3 describes the security mechanisms that can be implemented for each data retrieval interface. For device retrieval, issuer data authentication, session encryption and mdoc authentication shall be implemented. mdoc reader authentication is optional for device retrieval and TLS client authentication is optional for server retrieval. For server retrieval, Transport Layer Security (TLS), and JSON Web Signature (JWS) shall be implemented.

Table 4 describes security goals and mechanisms for different data retrieval methods.

When server retrieval is used, the mDL should use one-time server retrieval tokens.

The certificate and CRL profile requirements in Annex B shall be applied.

All certificates issued by an IACA or another CA shall be validated according to 9.3.3.

An mDL reader needs access to the issuing authority's certificate authority (IACA) root certificate to verify issuer data authentication, verify the JWS and perform TLS. One optional method to get access to these certificates is described in Annex C.

See Annex E for additional information on privacy and security.

Table 3 — Security mechanisms

Data retrieval method	Security mechanisms	Reference
Device retrieval	Session encryption	9.1.1
	Issuer data authentication	9.1.2
	mdoc authentication	9.1.3
	mdoc reader authentication	9.1.4
[a] Only applicable if the server retrieval token is transferred using device retrieval.		

Table 3 *(continued)*

Data retrieval method	Security mechanisms	Reference
Server retrieval	TLS	9.2.1
	JWS	9.2.2
	mdoc authentication [a]	9.1.3
[a] Only applicable if the server retrieval token is transferred using device retrieval.		

Table 4 — Security goals and mechanisms

Security goal	Functionality	Device retrieval	Server retrieval
Protection against forgery	Authenticate the origin of mDL data	Issuer data authentication	JWS
	Verify mDL data has not changed from issuing authority	Issuer data authentication	JWS
	Verify how up to date the mDL data is	Issuer data authentication	JWS
Protection against cloning	Protect against cloning of mDL/ binding mDL data to a specific device	mdoc authentication	mdoc authentication[a]
Protection against eavesdropping	Preserve confidentiality of mDL data	Session encryption	TLS
	Prevent unnoticed alteration of communication	Session encryption mdoc authentication	TLS
Protection against unauthorized access	Prevent unauthorized access of mDL data	Close-range device engagement with session encryption	Close-range device engagement (with session encryption)[a]
	Prevent unauthorized access of mDL data	mdoc reader authentication[b]	TLS client authentication[b]
[a] Only applicable if the server retrieval token is transferred using device retrieval. [b] This is an optional method.			

7 mDL data model

7.1 mDL document type and namespace

`DocType` and `NameSpace` are used to encapsulate the document type and the space in which the data elements are defined. The two concepts are further defined in 8.3.1.

The document type for an mDL document shall be "org.iso.18013.5.1.mDL". The number "1" in the document type might be increased in future versions of the standard.

NOTE 1 The document type field follows the following general format: [Reverse Domain].[Domain Specific Extension]. The reverse domain (org.iso) was selected to avoid collisions. This approach can be used to define other doctypes.

If the mDL reader wants to retrieve an mDL, the `DocType` field in the device retrieval mdoc request or server retrieval mdoc request shall contain the mDL document type. The use of any other value for the `DocType` field in the device retrieval mdoc request or server retrieval mdoc request is beyond the scope of this document.

The namespace for mDL data defined in 7.2 shall be "org.iso.18013.5.1". The number "1" in the namespace might be increased in future versions of the standard. Within this namespace, only data

elements defined in 7.2 and 8.2.1.2 may be used. To accommodate domestic data, an issuing authority may define its own namespace, as described in 7.2.8.

NOTE 2 The namespace field follows the following general format: [Reverse Domain].[Domain Specific Extension].

7.2 mDL data

7.2.1 Overview

The mDL data elements shall be as defined in Table 5 and belong to namespace "org.iso.18013.5.1", see 7.1.

— The "Identifier" column is used for `DataElementIdentifier` in the device retrieval mdoc request or server retrieval mdoc request (see 8.3.1).

— The "Presence" column indicates whether the presence of the element on an mDL is mandatory (M), or optional (O).

NOTE 1 This does not mean that granting access to these elements to an mDL reader is mandatory.

— The "Encoding format" column indicates how the data elements shall be encoded. "tstr", "uint", "bstr", "bool" and "tdate" are CDDL representation types as defined in RFC 8610. This document specifies "full-date" as `full-date = #6.1004(tstr)`, where tag 1004 is specified in RFC 8943.

— In accordance with RFC 7049, section 2.4.1, a tdate data item shall contain a date-time string as specified in RFC 3339. In accordance with RFC 8943, a `full-date` data item shall contain a full-date string as specified in RFC 3339.

— If data elements are encoded with JSON for the server retrieval methods, the data elements shall be encoded as specified in RFC 7049, section 4.1.

— The following requirements shall apply to the representation of dates in mDL data elements, unless otherwise indicated:

  — fraction of seconds shall not be used;

  — no local offset from UTC shall be used, as indicated by setting the `time-offset` defined in RFC 3339 to "Z".

Table 5 — Data elements

Identifier	Meaning	Definition	Presence	Encoding format
family_name	Family name	Last name, surname, or primary identifier, of the mDL holder. The value shall only use latin1[b] characters and shall have a maximum length of 150 characters.	M	tstr
given_name	Given names	First name(s), other name(s), or secondary identifier, of the mDL holder. The value shall only use latin1[b] characters and shall have a maximum length of 150 characters.	M	tstr
birth_date	Date of birth	Day, month and year on which the mDL holder was born. If unknown, approximate date of birth	M	full-date
issue_date	Date of issue	Date when mDL was issued	M	tdate or full-date
expiry_date	Date of expiry	Date when mDL expires	M	tdate or full-date
issuing_country	Issuing country	Alpha-2 country code, as defined in ISO 3166-1, of the issuing authority's country or territory	M	tstr
issuing_authority	Issuing authority	Issuing authority name. The value shall only use latin1[b] characters and shall have a maximum length of 150 characters.	M	tstr
document_number	Licence number	The number assigned or calculated by the issuing authority. The value shall only use latin1[b] characters and shall have a maximum length of 150 characters.	M	tstr
portrait	Portrait of mDL holder	A reproduction of the mDL holder's portrait. See 7.2.2	M	bstr
driving_privileges	Categories of vehicles/ restrictions/ conditions	Driving privileges of the mDL holder. See 7.2.4	M	See 7.2.4

Key

Presence:

M mandatory

O optional

NOTE 1 The 'UN Distinguishing sign' element is added for purposes of the UN conventions[26][27] on driving licences.

NOTE 2 The 'sex' element is equal to the 'gender' fields in ISO/IEC 18013-1 and ISO/IEC 18013-2.

NOTE 3 The 'issuing jurisdiction' element can be used in cases where the issuing jurisdiction is different from the issuing authority or the issuing country.

[a] The mDL reader can convert to the local unit of measurement.

[b] Latin1 shall be as defined in ISO/IEC 8859-1 as Latin alphabet No. 1.

Table 5 *(continued)*

Identifier	Meaning	Definition	Presence	Encoding format
un_distinguishing_sign	UN distinguishing sign	Distinguishing sign of the issuing country according to ISO/IEC 18013-1:2018, Annex F. If no applicable distinguishing sign is available in ISO/IEC 18013-1, an IA may use an empty identifier or another identifier by which it is internationally recognized. In this case the IA should ensure there is no collision with other IA's.	M	tstr
administrative_number	Administrative number	An audit control number assigned by the issuing authority. The value shall only use latin1[b] characters and shall have a maximum length of 150 characters.	O	tstr
sex	Sex	mDL holder's sex using values as defined in ISO/IEC 5218.	O	uint
height	Height (cm) [a]	mDL holder's height in centimetres	O	uint
weight	Weight (kg) [a]	mDL holder's weight in kilograms	O	uint
eye_colour	Eye colour	mDL holder's eye colour. The value shall be one of the following: "black", "blue", "brown", "dichromatic", "grey", "green", "hazel", "maroon", "pink", "unknown".	O	tstr
hair_colour	Hair colour	mDL holder's hair colour. The value shall be one of the following: "bald", "black", "blond", "brown", "grey", "red", "auburn", "sandy", "white", "unknown".	O	tstr
birth_place	Place of birth	Country and municipality or state/province where the mDL holder was born. The value shall only use latin1[b] characters and shall have a maximum length of 150 characters.	O	tstr
resident_address	Permanent place of residence	The place where the mDL holder resides and/or may be contacted (street/house number, municipality etc.). The value shall only use latin1[b] characters and shall have a maximum length of 150 characters.	O	tstr
portrait_capture_date	Portrait image timestamp	Date when portrait was taken	O	tdate

Key

Presence:

M mandatory

O optional

NOTE 1 The 'UN Distinguishing sign' element is added for purposes of the UN conventions[26][27] on driving licences.

NOTE 2 The 'sex' element is equal to the 'gender' fields in ISO/IEC 18013-1 and ISO/IEC 18013-2.

NOTE 3 The 'issuing jurisdiction' element can be used in cases where the issuing jurisdiction is different from the issuing authority or the issuing country.

[a] The mDL reader can convert to the local unit of measurement.

[b] Latin1 shall be as defined in ISO/IEC 8859-1 as Latin alphabet No. 1.

Table 5 *(continued)*

Identifier	Meaning	Definition	Presence	Encoding format
age_in_years	Age attestation: How old are you (in years)?	The age of the mDL holder	O	uint
age_birth_year	Age attestation: In what year were you born?	The year when the mDL holder was born	O	uint
age_over_NN	Age attestation: Nearest "true" attestation above request	See 7.2.5	O	bool
issuing_jurisdiction	Issuing jurisdiction	Country subdivision code of the jurisdiction that issued the mDL as defined in ISO 3166-2:2020, Clause 8. The first part of the code shall be the same as the value for issuing_country.	O	tstr
nationality	Nationality	Nationality of the mDL holder as a two letter country code (alpha-2 code) defined in ISO 3166-1	O	tstr
resident_city	Resident city	The city where the mDL holder lives. The value shall only use latin1[b] characters and shall have a maximum length of 150 characters.	O	tstr
resident_state	Resident state/province/district	The state/province/district where the mDL holder lives. The value shall only use latin1[b] characters and shall have a maximum length of 150 characters.	O	tstr
resident_postal_code	Resident postal code	The postal code of the mDL holder. The value shall only use latin1[b] characters and shall have a maximum length of 150 characters.	O	tstr
resident_country	Resident country	The country where the mDL holder lives as a two letter country code (alpha-2 code) defined in ISO 3166-1.	O	tstr
biometric_template_xx	Biometric template XX	See 7.2.6	O	bstr
family_name_national_character	Family name in national characters	The family name of the mDL holder using full UTF-8 character set.	O	tstr

Key

Presence:

M mandatory

O optional

NOTE 1 The 'UN Distinguishing sign' element is added for purposes of the UN conventions[26][27] on driving licences.

NOTE 2 The 'sex' element is equal to the 'gender' fields in ISO/IEC 18013-1 and ISO/IEC 18013-2.

NOTE 3 The 'issuing jurisdiction' element can be used in cases where the issuing jurisdiction is different from the issuing authority or the issuing country.

[a] The mDL reader can convert to the local unit of measurement.

[b] Latin1 shall be as defined in ISO/IEC 8859-1 as Latin alphabet No. 1.

Table 5 *(continued)*

Identifier	Meaning	Definition	Presence	Encoding format
given_name_national_character	Given name in national characters	The given name of the mDL holder using full UTF-8 character set.	O	tstr
signature_usual_mark	Signature / usual mark	Image of the signature or usual mark of the mDL holder, see 7.2.7	O	bstr

Key

Presence:

M mandatory

O optional

NOTE 1 The 'UN Distinguishing sign' element is added for purposes of the UN conventions[26][27] on driving licences.

NOTE 2 The 'sex' element is equal to the 'gender' fields in ISO/IEC 18013-1 and ISO/IEC 18013-2.

NOTE 3 The 'issuing jurisdiction' element can be used in cases where the issuing jurisdiction is different from the issuing authority or the issuing country.

[a] The mDL reader can convert to the local unit of measurement.

[b] Latin1 shall be as defined in ISO/IEC 8859-1 as Latin alphabet No. 1.

If any data element is returned to the mDL reader, then the information necessary to verify that the person presenting the mDL is the mDL holder shall be returned to the mDL reader if that information is requested.

NOTE 2 From the set of data elements with mandatory presence in column "Presence" from Table 5, the portrait of the mDL holder is the only data item for verifying that the person presenting the mDL is the mDL holder.

An mDL may require mdoc reader authentication (see 9.1.4) before releasing data elements not marked as mandatory in Table 5. An mDL shall not require mdoc reader authentication as a precondition for the release of any of the mandatory data elements. An mDL may offer functionality to the mDL holder to pre-authorize the release of mandatory data elements selected by the mDL holder to mDL readers using mdoc reader authentication.

NOTE 3 The intention of these requirements is that the mDL holder is always able to use the mDL as a driving licence if the mDL holder chooses to do that, including if an mDL reader does not use mdoc reader authentication.

If device retrieval is used, the "issue_date" data element shall not be after the `validFrom` element as defined in 9.1.2.4.

If server retrieval is used, the "issue_date" data element shall not be after the `iat` claim as described in 8.3.2.2.2.2 and 8.3.3.2.2.

If the mDL reader retrieved the "issuing_country" data element, it shall verify that the value of that element matches the `countryName` element in the subject field within the DS certificate or the JWS signer certificate (see Annex B).

If device retrieval is used, if the mDL reader retrieved the "issuing_jurisdiction" data element, it shall verify that the value of that element matches the `stateOrProvinceName` element in the subject field within the DS certificate (see Annex B). This is only required if the `stateOrProvinceName` element is present in the DS certificate.

If server retrieval is used, if the mDL reader retrieved the "issuing_jurisdiction" data element, it shall verify that the value of that element matches the `stateOrProvinceName` element in the subject field within the JWS signer certificate (see Annex B). This is only required if the `stateOrProvinceName` element is present in the JWS signer certificate."

If device retrieval is used, any element in Table 5 that is returned by the mDL shall be returned as part of the `IssuerSignedItems`. Domestic data elements (see 7.2.8) may be returned in either `IssuerSignedItems` or `DeviceSignedItems`.

NOTE 4 Issuing authorities have the responsibility to ensure that controls to prevent the unintended release of mDL data are implemented, including providing mDL holders with the means to control what data is released.

7.2.2 Portrait of mDL holder

The portrait of mDL holder consists of one portrait image and shall follow the requirements on the face image as specified in ISO/IEC 18013-2:2020, Annex D. One of the following image formats shall be used: JPEG or JPEG2000. The image data shall be encoded as binary data.

7.2.3 Issuing authority

The issuing authority element identifies the administrative authority entitled to issue the driving licence, or the issuing country if separate licensing authorities have not been authorized. The issuing authority element is represented by a string.

NOTE The contents of this field correspond to the contents of the issuing authority element on the IDL, and can indicate a local, regional, or national organisation. Note that, like in ISO/IEC 18013-1, the term issuing authority can also refer to a central government agency, acting on behalf of multiple local or regional issuing authorities.

7.2.4 Categories of vehicles/restrictions/conditions

The categories of vehicles/restrictions/conditions contain information describing the driving privileges of the mDL holder. The definition of the elements in the `DrivingPrivilege` structure can be found in ISO/IEC 18013-1:2018, Clause 5. The possible values for the elements are defined in ISO/IEC 18013-1:2018, Annex B and ISO/IEC 18013-2:2020, Annex A.

NOTE 1 Regardless of driving privileges' status, other mDL data elements can be used for identification purposes as described by this document.

The driving privileges structure shall be encoded as CBOR for device retrieval and JSON for server retrieval and shall be formatted as follows:

```
DrivingPrivileges = [
    * DrivingPrivilege
]

DrivingPrivilege = {
    "vehicle_category_code" : tstr      ; Vehicle category code as per ISO/IEC 18013-1
Annex B
    ? "issue_date" : full-date          ; Date of issue encoded as full-date
    ? "expiry_date" : full-date         ; Date of expiry encoded as full-date
    ? "codes" : [+Code]                 ; Array of code info
}

Code = {
    "code": tstr                        ; Code as per ISO/IEC 18013-2 Annex A
    ? "sign": tstr                      ; Sign as per ISO/IEC 18013-2 Annex A
    ? "value": tstr                     ; Value as per ISO/IEC 18013-2 Annex A
}
```

NOTE 2 The `DrivingPrivileges` structure can be an empty array.

Full-date is defined in 7.2.1.

An example can be found in D.2.1.

7.2.5 Age attestation: nearest “true” attestation above request

This set of elements is used to convey to an mDL verifier, in a data-minimized fashion, if the mDL holder is as old or older than a specified age, or if the mDL holder is younger than a specified age. To achieve this, the mDL contains age attestation identifiers. An age attestation identifier has the format age_over_NN where NN is a value from 00 to 99. The value of an age attestation identifier can be TRUE or FALSE.

If an mDL verifier includes age_over_NN in a request, it has the meaning of “provide the nearest age attestation equal to or larger than NN with value TRUE, or smaller than NN with value FALSE”. More specifically, after receiving an age_over_NN request, the logic to determine the appropriate response shall be equivalent to the following.

1. For all age attestations of the form age_over_nn stored on the mDL, consider all the attestations with value TRUE. From among these attestations, check if an attestation exists where nn is equal to or larger than NN. If one and only one such attestation exists, this is the response. If more than one such attestation exists, the response shall be the attestation with the smallest difference between nn and NN.

2. If step 1 does not produce a response, for all age attestations of the form age_over_nn stored on the mDL, consider all the attestations with value FALSE. From among these attestations, check if an attestation exists where nn is equal to or smaller than NN. If one and only one such attestation exists, this is the response. If more than one such attestation exists, the response shall be the attestation with the smallest difference between NN and nn.

3. If step 2 does not produce a response, no age_over_nn data element shall be returned.

In case of device retrieval, the value of an age_over_NN data element shall be calculated by the issuing authority infrastructure to be valid at the value of the timestamp in the `validFrom` element in the MSO from 9.1.2.4.

In case of server retrieval, the value of an age_over_NN data element shall be valid at the value of the `iat` timestamp as defined in 8.3.2.2.2.2 and 8.3.3.2.2.

During a single data retrieval phase (see Figure 3), an mDL reader shall not request more than two age_over_NN data elements, and an mDL or issuing authority infrastructure should not return more than two age_over_NN data elements.

Examples of requests and responses and their meaning can be found in D.2.2.

NOTE 1 Including more rather than less age_over_NN statements in an mDL has the following consequences, both of which are beneficial from a privacy perspective.

1. If an mDL does not respond to an age_over_NN request, the most likely next step of a verifier will be to ask for the mDL holder’s date of birth. This would be the only remaining way in which the original business question could be answered. Including additional age_over_NN statements decreases the probability of a no response.

2. Given the question “are you older than 18?”, the response “I am older than 21” and the response “I am older than 60” would both answer the question. However, the response “I am older than 21” is considered better from a privacy point of view, since the statement is closer to the request. Including additional age_over_NN statements will on average decrease the difference between the requested age and the response age.

Including more rather than less age_over_NN statements in a mDL on the other hand could allow a verifier to gain a more accurate estimate of a person’s true age, if for whatever reason such a verifier chooses not to ask directly for the mDL holder’s date of birth. This would require repeated requests.

NOTE 2 A request can include two age_over_NN statements to support effectively asking whether the age of the mDL holder falls within a certain range.

NOTE 3 It is possible that an mDL does not have an age_over_NN element available to respond to the request from the mDL reader. If an mDL reader does not receive an age_over_NN response, the mDL reader has the option of making another request with different data elements that fulfil the business requirements that it has. Examples of these are 'age_in_years' and ‘birth_date’.

7.2.6 Biometric template

This element contains optional facial, fingerprint, iris, or other biometric information of the mDL holder. Biometric information is encoded according to the biometric template defined in ISO/IEC 18013-2:2020, Table 7 and C.4.7. However, the first tag in ISO/IEC 18013-2:2020, Table C.11 ('75' for Facial, '63' for Finger, '76' for Iris, etc) shall be omitted. The value of the biometric_template_xx data element shall therefore start with tag '7F 61'. A biometric template identifier has the format biometric_template_xx where xx shall be replaced with the corresponding "Abstract value name" found in ISO/IEC 19785-3:2020, Table 7, according to the following convention: capitalized characters are replaced with their lowercase equivalent and spaces or non-alphanumeric characters are replaced by underscores (_).

EXAMPLE The "FACE" template corresponds to "biometric_template_face" and the "SIGNATURE/SIGN" template corresponds to "biometric_template_signature_sign".

If the biometric_template_face is used, the biometric data block (tag '5F 2E') shall contain a JPEG or JPEG2000 format, as specified in 7.2.2.

7.2.7 Signature or usual mark

The signature or usual mark of the mDL holder consists of one image. One of the following image formats shall be used: JPEG or JPEG2000. The image data shall be encoded as binary data.

7.2.8 Domestic data elements

Domestic data are data elements which are not specified in this document. An issuing authority may specify its own data elements within its namespace. Since the namespace for mDL data in this document is "org.iso.18013.5.1", the issuing authority infrastructure should use country-specific or issuer-specific namespaces by appending the ISO 3166-1 alpha-2 country code or the ISO 3166-2 region code after a period.

EXAMPLE The United States namespace is "org.iso.18013.5.1.US" and the Iowa namespace is "org.iso.18013.5.1.US-IA".

7.3 Country codes

ISO 3166-1 shall be used as a source for country code identifiers. If no applicable country code is available in ISO 3166-1, an IA may use one of the user-assigned country code elements, as indicated in ISO 3166-1 or another identifier by which it is internationally recognized. In these cases, the IA should ensure there is no collision with other IAs. These provisions apply to all occurrences of country code identifiers in this document unless stated otherwise.

8 Transaction

8.1 Encoding of data structures and data elements

In this document CDDL (Concise Data Definition Language) as specified in RFC 8610 is used to express CBOR and JSON-encoded data structures.

CBOR structures shall be encoded according to RFC 7049. JSON structures shall be encoded according to RFC 8259.

RFC 7049, section 3.9 describes four rules for canonical CBOR. Three of those rules shall be implemented for all CBOR structures as follows:

— integers (major types 0 and 1) shall be as small as possible;

— the expression of lengths in major types 2 through 5 shall be as short as possible;

— indefinite-length items shall be made into definite-length items.

The fourth rule regarding sorting of map keys is not required. Furthermore, maps (major type 5) shall not have multiple entries with the same key.

Because canonical map ordering is not required, all CBOR maps that are used in a cryptographic operation are communicated in a tagged CBOR bytestring. For any cryptographic operation, an mdoc, mdoc reader or issuing authority infrastructure shall use these bytestrings as they were sent or received, without attempting to re-create them from the underlying maps.

EXAMPLE A data structure `DataItem` that is to be used in a cryptographic operation is communicated in a structure `DataItemBytes`, specified as follows:

```
DataItemBytes = #6.24(bstr .cbor DataItem)
```

The CDDL in this example is defined in RFC 8610, section 3.6 and expresses a tagged data item (major type 6). As specified in RFC 7049, section 2.4, tag value 24 indicates that the content of the CBOR bstr following the tag is itself a CBOR data item. The .cbor control operator indicates that this data item is in fact a `DataItem`.

When processing a data structure, an mdoc, mdoc reader or issuing authority infrastructure shall ignore any value that is specified as RFU in this document.

Whenever data structures in this document use a version element that is encoded as a string, their contents follow the format of 'major version number'.'minor version number'. A major version number shall be incremented by 1 when any backwards incompatible changes are introduced. In a future version of this document. A minor version number shall be incremented by 1 when new, but backwards compatible functionality is introduced. A minor version number shall be reset to 0 if the major version number is incremented. An mdoc, mdoc reader or issuing authority infrastructure shall not give an error and continue a transaction if it receives a data structure having a known major version number but with an unknown minor version number.

8.2 Device engagement

8.2.1 Device engagement information

8.2.1.1 Device engagement structure

The device engagement structure contains information to perform device engagement. The device engagement structure shall be CBOR encoded and formatted as follows:

```
DeviceEngagement =
{
    0: tstr,              ; Version
    1: Security,
    ? 2: DeviceRetrievalMethods,  ; Is absent if NFC is used for device engagement
    ? 3: ServerRetrievalMethods,
    ? 4: ProtocolInfo,
    * int => any
}

Security = [
    int,                ; Cipher suite identifier
    EDeviceKeyBytes
]

DeviceRetrievalMethods = [
    + DeviceRetrievalMethod
]

ServerRetrievalMethods = {
    ?"webApi" : WebApi,
    ?"oidc" : Oidc
}

ProtocolInfo = any      ; The use of ProtocolInfo is RFU
```

```
DeviceRetrievalMethod = [
    uint,               ; Type
    uint,               ; Version
    RetrievalOptions    ; Specific option(s) to the type of retrieval method
]

RetrievalOptions = WifiOptions / BleOptions / NfcOptions / any  ; The any option is RFU
```

The device engagement structure contains the following key-value pairs.

0. `Version`: the version of the device engagement structure, in the current version of this document its value shall be "1.0".

1. `Security`: an array that contains two mandatory elements. The first element is the cipher suite identifier, defined in 9.1.5.2. The second element is `EdeviceKeyBytes`, defined in 9.1.1.4.

2. `DeviceRetrievalMethods`: an array that shall contain one or more `DeviceRetrievalMethod` arrays when performing device engagement using the QR code. When using NFC to perform device engagement, the `DeviceRetrievalMethods` array shall be absent, because the data retrieval methods supported by the mdoc are specified in the Alternative Carrier Records, as specified in 8.2.2.1. `DeviceRetrievalMethods` lists the device retrieval methods supported by the mdoc. A `DeviceRetrievalMethod` array holds two mandatory values (type and version). The first element defines the type and the second element the version for the transfer method. The `RetrievalOptions` element may contain extra info for each connection. The values for the different `RetrievalOptions` are defined in 8.2.2.3.

3. `ServerRetrievalMethods`: a map that contains optional information on the server retrieval methods supported by the mdoc. The values for the different server retrieval methods are defined in 8.2.1.2. If the `ServerRetrievalMethods` is present, the map shall contain `webApi`, `oidc`, or both.

4. `ProtocolInfo`: to ensure the DeviceEngagement remains applicable to future solutions and updates, the contents of the `ProtocolInfo` are RFU in consideration for defining protocol information.

The following requirements apply for additional key-value pairs within the device engagement structure: positive integers for keys are RFU. An application-specific extension shall use a negative integer for the key. An mdoc or mdoc reader shall ignore any key-value pairs with a negative key value that it is not able to interpret.

An example of a device engagement structure can be found in D.3.1.

8.2.1.2 Server retrieval information

Setting up server retrieval requires the mdoc to provide the server retrieval information to the mdoc reader to facilitate the mdoc reader's use of the server retrieval method.

The server retrieval information can be transferred as part of the device engagement structure or as a data element during device retrieval.

— If the server retrieval information is transferred as part of device engagement, the `Oidc`, `WebApi` or both structures shall be transferred as part of `ServerRetrievalMethods` in the device engagement structure (see 8.2.1.1).

— If the server retrieval information is transferred as part of device retrieval, the `Oidc`, `WebApi` or both structures shall be transferred as data elements in the device retrieval mdoc response as defined in 8.3.2.1.2.2. They shall be returned in either `IssuerSignedItems` or `DeviceSignedItems`. The `Oidc` structure shall have the data element identifier "oidc_info". The `WebApi` structure shall have the data element identifier "webapi_info". The namespace of these data elements shall be "org.iso.18013.5.1".

If the mdoc reader wants to use server retrieval, it can request the "webapi_info" or "oidc_info" data elements as a part of the device retrieval mdoc request. The mdoc reader may also do this if the mdoc already included the server retrieval information in the device engagement structure.

If the server retrieval information is transferred during device retrieval, its authenticity is protected by either issuer data authentication (9.1.2) or mdoc authentication (9.1.3). If the server retrieval information is transferred in the device engagement structure, it is not protected. Therefore, the IA is responsible for the server retrieval token sent in the device engagement structure to authorize the request as part of the transaction by or on behalf of the mdoc holder.

The structures that contain the server retrieval information shall be encoded as CBOR and formatted as follows:

```
Oidc = [
    uint,    ; Version
    tstr,    ; Issuer URL
    tstr     ; Server retrieval token
]

WebApi = [
    uint,    ; Version
    tstr,    ; Issuer URL
    tstr     ; Server retrieval token
]
```

Both arrays consist of three fields: the version, the issuer URL, and the server retrieval token. The version indicates the version of the transfer methods, in the current version of this document, its value shall be 1 for both OIDC and WebAPI. The URL and the server retrieval token field are further defined in 8.3.2.2.

8.2.2 Device engagement transmission technology

8.2.2.1 Device engagement using NFC

Device engagement using NFC shall follow the Connection Handover protocol as defined by NFC Forum, *Connection Handover (CH) Technical Specification, Version 1.5*. Only Reader/Writer mode using the Type 4 Tag shall be used. The Connection Handover protocol shall be initiated by the mdoc reader. The mdoc reader shall take the role of Handover Requester. The mdoc shall be the NFC Tag Device and the mdoc reader shall be the NFC Reader Device. The mdoc shall use either Static Handover or Negotiated Handover.

When Static Handover is used, the Handover Select Message shall be retrieved by the mdoc reader from the mdoc in a Type 4 Tag and shall contain at least one Alternative Carrier Record. Each Alternative Carrier Record shall indicate a device retrieval method the mdoc supports. An mdoc reader shall select one of the transmission technologies from the ones provided in the Alternative Carrier Records.

When Negotiated Handover is used, the mdoc shall include the "urn:nfc:sn:handover" service in a Service Parameter record in the initial NDEF message provided to the mdoc reader. The mdoc reader shall send a Handover Request Message to the mdoc after the it has selected this service. The Handover Request Message shall contain an Alternative Carrier Record for each alternative carrier that is supported by the mdoc reader. The mdoc confirms the handover by providing a Handover Select Message containing exactly one selected alternative carrier.

NOTE Use of Negotiated Handover for device engagement allows negotiation of transfer methods. For BLE and Wi-Fi Aware, it additionally allows negotiation of keys used by the transmission layer. This can provide better user experience and security of data transmission.

For the BLE device retrieval transmission technology, the contents of the Alternative Carrier Record and Carrier Configuration Record(s) shall comply with 8.3.3.1.1.2.

For the NFC device retrieval transmission technology, the contents of the Alternative Carrier Record and Carrier Configuration Record(s) shall comply with 8.2.2.2.

For the Wi-Fi Aware device retrieval transmission technology, the contents of the Alternative Carrier Record and Carrier Configuration Record(s) shall comply with the Wi-Fi Alliance Neighbor Awareness Networking Specification, version 3.1, section 12.

The `DeviceEngagement` structure as defined in 8.2.1.1 shall be transferred from the mdoc to the mdoc reader as part of an auxiliary data record of the Handover Select Message with the type "iso.org:18013:deviceengagement" and the ID reference "mdoc", with type name format NFC Forum external type (0x04). For each ac record, "Auxiliary Data Reference" points to the NDEF record which contains the device engagement structure.

When Negotiated Handover is used, the mdoc reader may sent a `ReaderEngagement` structure to the mdoc as part of an auxiliary data record of the Handover Request Message with the type "iso.org:18013:readerengagement" and the ID reference "mdocreader", with type name format NFC Forum external type (0x04), to provide information from the mdoc reader to the mdoc.

The reader engagement structure shall be CBOR encoded and formatted as follows:

```
ReaderEngagement =
{
    0: tstr,              ; Version
    * int => any
}
```

The reader engagement structure contains the following key-value pairs:

0. `Version`: the version of the reader engagement structure. In the current version of this document its value shall be "1.0".

This document does not define any other key-value pairs for use in `ReaderEngagement`. The following requirements apply for additional key-value pairs within the reader engagement structure: positive integers for keys are RFU. An application-specific extension shall use a negative integer for the key. An mdoc or mdoc reader shall ignore any key-value pairs with a negative key value that it is not able to interpret.

8.2.2.2 Alternative Carrier Record for device retrieval using NFC

The Alternative Carrier Record for the NFC device retrieval transmission technology shall reference the Carrier Configuration Record with the ID reference "nfc".

The Carrier Configuration Record for NFC device retrieval transmission technology shall have the type "iso.org:18013:nfc" and the ID reference "nfc". The binary content of the Carrier Configuration Record shall be encoded according to Table 6.

Table 6 — Binary content of the Carrier Configuration Record for NFC device retrieval

Field	Size (Octet)	Sub-Field	Value	Presence
Version	1		0x01	Mandatory
Maximum command data length	1	Length	See below	Conditional
	1	Data type	0x01	
	Variable	Maximum command data length	See 8.3.3.1.2	
Maximum response data length	1	Length	See below	Conditional
	1	Data type	0x02	
	Variable	Maximum response data length	See 8.3.3.1.2	

The value of the version field shall be the mdoc NFC Connection Handover Version encoded as an unsigned integer. This is a mandatory field.

For each field, the value of the length sub-field shall be the sum of the length of the other two sub-fields within that field, encoded as an unsigned integer.

The value of the maximum command data length field shall be the maximum length of command data field supported by the mdoc, encoded as an unsigned integer. This field is further defined in 8.3.3.1.2. This field is mandatory for the mdoc and shall not be used by the mdoc reader.

The value of the maximum response data length field shall be the maximum length of response data field supported by the mdoc, encoded as an unsigned integer. This field is further defined in 8.3.3.1.2. This field is mandatory for the mdoc and shall not be used by the mdoc reader.

8.2.2.3 Device engagement using QR code

If QR code is used for device engagement, the device engagement structure shall be transmitted as a barcode compliant with ISO/IEC 18004. The QR code shall contain a URI with "mdoc:" as scheme and the `DeviceEngagement` structure specified in 8.2.1.1 encoded using base64url-without-padding, according to RFC 4648, as path.

NOTE The requirements above result in the content of the QR code as "mdoc:" followed by the base64url-without-padding encoded device engagement structure.

An mdoc reader shall select one of the transmission technologies from the ones provided in the device engagement structure.

Table 7 defines the values that shall be used for the `DeviceRetrievalMethod` structure. Any other `DeviceRetrievalMethod` type is RFU.

Table 7 — DeviceRetrievalMethod parameters

	NFC	BLE	Wi-Fi Aware
type	1	2	3
version	1	1	1
options	`NfcOptions`	`BleOptions`	`WifiOptions`

The `WifiOptions`, `BleOptions` and `NfcOptions` structures shall be formatted as follows. Other `RetrievalOptions` are RFU.

```
WifiOptions = {
    ? 0: tstr,       ; Pass-phrase Info Pass-phrase
    ? 1: uint,       ; Channel Info Operating Class
    ? 2: uint,       ; Channel Info Channel Number
    ? 3: bstr        ; Band Info Supported Bands
}

BleOptions = {
    0 : bool,        ; Indicates support for mdoc peripheral server mode
    1 : bool,        ; Indicates support for mdoc central client mode
    ? 10 : bstr,     ; UUID for mdoc peripheral server mode
    ? 11 : bstr,     ; UUID for mdoc client central mode
    ? 20 : bstr          ; mdoc BLE Device Address for mdoc peripheral server mode
}

NfcOptions = {
    0 : uint,        ; Maximum length of command data field
    1 : uint         ; Maximum length of response data field
}
```

The contents of these fields are further defined in 8.3.3.1.1, 8.3.3.1.2, and 8.3.3.1.3.

8.2.3 Device engagement time-out

If the mdoc reader implements a time-out for the time between the transaction initialization and receiving device engagement data, the time-out should be no less than 30 seconds. The mdoc verifier may terminate the session at any time.

If the mdoc or mdoc reader implements a time-out for the time between receiving or sending device engagement data and sending or receiving the session establishment message (see 9.1.1.4), the time-out should be no less than 30 seconds. The mdoc holder and mdoc verifier may terminate the session at any time.

8.3 Data retrieval

8.3.1 Data model

The document type that is requested by an mdoc reader or returned by an mdoc or IA infrastructure is encapsulated in the `DocType` parameter.

NOTE 1 There is no requirement for the `DocType` format. An approach to avoid collisions is to use the following general format: [Reverse Domain].[Domain Specific Extension].

The `NameSpace` parameter provides the namespace within which the data elements requested by the mdoc reader or returned by the mdoc are defined. A document may contain data elements from multiple namespaces. The meaning of data elements is dependent on their namespace.

NOTE 2 There is no requirement for the `NameSpace` format. An approach to avoid collisions is to use the following general format: [Reverse Domain].[Domain Specific Extension].

The CDDL definitions for `DocType`, `NameSpace`, `DataElementIdentifier` and `DataElementValue` are common across different data retrieval methods and the applicable security mechanisms. The following CDDL definitions shall be applied to the CDDL structures defined in Clause 8 and Clause 9:

```
DocType = tstr
NameSpace = tstr
DataElementIdentifier = tstr     ; Data element identifier
DataElementValue = any           ; Data element value
```

8.3.2 Data retrieval methods

8.3.2.1 Device retrieval

8.3.2.1.1 General

The mdoc request and mdoc response messages are defined in 8.3.2.1.2. They are encrypted and subsequently included in a session establishment or session data message, see 9.1.1. These session establishment and session data messages are then transferred using one of the device retrieval transmission methods specified in 8.3.3.1.1, 8.3.3.1.2, or 8.3.3.1.3.

8.3.2.1.2 Message structures

8.3.2.1.2.1 Device retrieval mdoc request

The device retrieval mdoc request structure shall be CBOR encoded and formatted as follows:

```
DeviceRequest = {
    "version" : tstr,                   ; Version of DeviceRequest structure
    "docRequests" : [+ DocRequest]      ; Requested documents
}

DocRequest = {
    "itemsRequest" : ItemsRequestBytes,
    ? "readerAuth" : ReaderAuth         ; mdoc reader authentication
}

ItemsRequestBytes = #6.24(bstr .cbor ItemsRequest)

ItemsRequest = {
    "docType" : DocType,                ; Document type requested
```

```
    "nameSpaces" : NameSpaces,
    ? "requestInfo" : {* tstr => any}    ; Additional information
}

NameSpaces = {
    + NameSpace => DataElements          ; Requested data elements for each NameSpace
}

DataElements = {
    + DataElementIdentifier => IntentToRetain    ; Requested data element identifiers
                                                  ; with intent to retain values
}

IntentToRetain = bool
```

`version` is the version for the `DeviceRequest` structure: in the current version of this document its value shall be "1.0". If other versions are specified in the future, the major version (see 8.1) of a `DeviceRequest` structure shall not be higher than the major version of the device engagement structure (see 8.2.1.1) communicated by the mdoc in the same transaction.

`docRequests` contains an array of all requested documents.

`ItemRequestBytes` contains the `ItemsRequest` structure as a tagged CBOR bytestring.

`ReaderAuth` is used for mdoc reader authentication as defined in 9.1.4.

`docType` is the requested document type.

`requestInfo` may be used by the mdoc reader to provide additional information. This document does not define any key-value pairs for use in `requestInfo`. An mdoc shall ignore any key-value pairs that it is not able to interpret.

`NameSpaces` contains the requested data elements and the namespace they belong to.

`DataElements` contains the requested data elements identified by their data element identifier. For each requested data element, the `IntentToRetain` variable indicates whether the mdoc verifier intends to retain the received data element. The mdoc verifier shall not retain any data, including digests and signatures, or derived data received from the mdoc, except for data elements for which the `IntentToRetain` flag was set to true in the request. To retain is defined as "to store for a period longer than necessary to conduct the transaction in realtime".

The mdoc shall ignore all unknown data elements in a device retrieval mdoc request when processing the request.

An example of a device retrieval mdoc request can be found in D.4.1.1.

8.3.2.1.2.2 Device retrieval mdoc response

The device retrieval mdoc response shall be CBOR encoded and formatted as follows:

```
DeviceResponse = {
    "version" : tstr,                    ; Version of the DeviceResponse structure
    ? "documents" : [+Document],         ; Returned documents
    ? "documentErrors": [+DocumentError]; For unreturned documents, optional error codes
    "status" : uint                      ; Status code
}

Document = {
    "docType" : DocType,                 ; Document type returned
    "issuerSigned" : IssuerSigned,       ; Returned data elements signed by the issuer
    "deviceSigned" : DeviceSigned,       ; Returned data elements signed by the mdoc
    ? "errors" : Errors
}

DocumentError = {
```

```
    DocType => ErrorCode                ; Error codes for unreturned documents
}

IssuerSigned = {
    ? "nameSpaces" : IssuerNameSpaces,  ; Returned data elements
    "issuerAuth" : IssuerAuth           ; Contains the mobile security object (MSO)
                                        ; for issuer data authentication
}


IssuerNameSpaces = {                    ; Returned data elements for each namespace
    + NameSpace => [ + IssuerSignedItemBytes ]
}

IssuerSignedItemBytes = #6.24(bstr .cbor IssuerSignedItem)

IssuerSignedItem = {
    "digestID" : uint,                  ; Digest ID for issuer data authentication
    "random" : bstr,                    ; Random value for issuer data authentication
    "elementIdentifier" : DataElementIdentifier,    ; Data element identifier
    "elementValue" : DataElementValue   ; Data element value
}

DeviceSigned = {
    "nameSpaces" : DeviceNameSpacesBytes,   ; Returned data elements
    "deviceAuth" : DeviceAuth               ; Contains the device authentication
                                            ; for mdoc authentication
}

DeviceNameSpacesBytes = #6.24(bstr .cbor DeviceNameSpaces)

DeviceNameSpaces = {
    * NameSpace => DeviceSignedItems    ; Returned data elements for each namespace
}

DeviceSignedItems = {
    + DataElementIdentifier => DataElementValue   ; Returned data element identifier and
value
}

DeviceAuth = {          ; Either signature or MAC for mdoc authentication
    "deviceSignature" : DeviceSignature //     ; "//" means or
    "deviceMac" : DeviceMac
}

Errors = {
    + NameSpace => ErrorItems               ; Error codes for each namespace
}

ErrorItems = {
    + DataElementIdentifier => ErrorCode    ; Error code per data element
}

ErrorCode = int                             ; Error code
```


`version` is the version for the `DeviceResponse` structure. In the current version of this document its value shall be "1.0". If other versions are specified in the future, the major version (see 8.1) of a `DeviceResponse` structure shall not be higher than the major version of the device engagement structure (see 8.2.1.1) communicated by the mdoc in the same transaction. The major version of a `DeviceResponse` structure shall also not be higher than the major version of the mdoc request (see 8.3.2.1.2.1) to which it is a response.

`documents` contains an array of all returned documents. `documentErrors` can contain error codes for documents that are not returned. `status` contains a status code according to 8.3.2.1.2.3.

In the `Document` structure, the document type of the returned document is indicated by the `docType` element. The document type shall match the document type as indicated in the issuer data

authentication (see 9.1.2) and mdoc authentication structures (see 9.1.3). `errors` can contain error codes for data elements that are not returned.

`IssuerSigned` contains the mobile security object for issuer data authentication and the data elements protected by issuer data authentication. `nameSpaces` contains the returned data elements as part of their corresponding namespaces.

The `IssuerAuth` structure is defined in 9.1.2.4.

Individual data elements are returned as `IssuerSignedItem`. The `digestID` and `random` are defined in 9.1.2.5. `elementIdentifier` is the data element identifier and `elementValue` the data element value. Each `IssuerSignedItem` is returned as part of the corresponding namespace in the `IssuerNameSpaces` structure. The mdoc shall not include two or more `IssuerSignedItem` elements with the same `DataElementIdentifier` in a single `NameSpace` and `Document`.

`DeviceSigned` contains the mdoc authentication structure and the data elements protected by mdoc authentication. `nameSpaces` contains the returned data elements as part of their corresponding namespaces. `nameSpaces` is a mandatory element because the element is authenticated using mdoc authentication. The `DeviceNameSpaces` structure can be an empty structure. The `DeviceAuth` structure contains either the `DeviceSignature` or the `DeviceMac` element, both are defined in 9.1.3.

`DeviceSignedItems` contains the data element identifiers and values. `DeviceSignedItems` is returned as part of the corresponding namespace in `DeviceNameSpaces`.

If the device retrieval mdoc response structure does not include some data element or document requested in the device retrieval mdoc request, an error code may be returned as part of the `documentErrors` or `errors` structures.

If present, `ErrorCode` shall contain an error code according to 8.3.2.1.2.3.

An example of a device retrieval mdoc response can be found in D.4.1.2.

8.3.2.1.2.3 Device retrieval mdoc response status and error codes

The `status` element shall contain one of the status codes in Table 8. If the mdoc returns a status code different from 0, it shall not return any documents.

Table 8 — Response status

Status code	Status message	Description	Actions required
0	OK	Normal processing. This status message shall be returned if no other status is returned	No specific action required
10	General error	The mdoc returns an error without any given reason.	The mdoc reader may inspect the problem. The mdoc reader may continue the transaction.
11	CBOR decoding error	The mdoc indicates an error during CBOR decoding that the data received is not valid CBOR. Returning this status code is optional.	The mdoc reader may inspect the problem. The mdoc reader may continue the transaction.
12	CBOR validation error	The mdoc indicates an error during CBOR validation, e.g. wrong CBOR structures. Returning this status code is optional.	The mdoc reader may inspect the problem. The mdoc reader may continue the transaction.

If present, the `ErrorCode` element shall contain an error code from Table 9. An `ErrorCode` is specific to the document or data element requested within a namespace.

Table 9 — Data handling error

Error code	Error code message	Description
0	Data not returned	The mdoc does not provide the requested document or data element without any given reason. This element may be used in all cases.
Other positive integers	See description	RFU
Negative integers	See description	These error codes may be used for application-specific purposes.

8.3.2.2 Server retrieval

8.3.2.2.1 General

Data retrieval using server retrieval shall make use of WebAPI or OIDC. Subclause 8.3.2.2.2 specifies the structures of server retrieval mdoc requests and server retrieval mdoc responses for WebAPI. The messages structures for WebAPI are transferred using the server data retrieval transmission method specified in 8.3.3.2.1. Subclause 8.3.3.2.2 specifies the server data retrieval transmission method for OIDC.

The mdoc is identified by the issuing authority infrastructure using the server retrieval token as provided by the mdoc reader (see 8.2.1.2). The information encoded in the server retrieval token is outside the scope of this document.

NOTE 1 The server retrieval token can contain information about which data is to be provided to the mdoc reader. The issuing authority infrastructure can use a separate interface with the mdoc to retrieve information about which data is to be provided to the mdoc reader.

NOTE 2 The issuing authority infrastructure is involved in each server retrieval-based transaction; therefore, the issuing authority knows when an mdoc is used and what data is shared. If tracking is a concern, the issuing authority can implement mitigating strategies to ensure the mdoc and the mdoc holder are not tracked.

8.3.2.2.2 WebAPI structures

8.3.2.2.2.1 Server retrieval mdoc request

The server retrieval mdoc request shall be JSON encoded and formatted as follows:

```
ServerRequest = {
    "version" : tstr,                       ; Version of the structure
    "token" : tstr,                         ; Server retrieval token
    "docRequests" : [+ ItemsRequest]        ; Requested documents
}

ItemsRequest = {
    "docType" : DocType,                    ; Document type requested
    "nameSpaces" : NameSpaces,
    ? "requestInfo" : {* tstr => any}       ; Additional information
}

NameSpaces = {
    + NameSpace => DataElements             ; Requested data elements for each NameSpace
}

DataElements = {
    + DataElementIdentifier => IntentToRetain    ; Requested data element identifiers
                                                  ; with intent to retain values
}

IntentToRetain = bool
```

`version` is the version for the `ServerRequest` structure: in the current version of this document its value shall be "1.0". If other versions are specified in the future, the major version (see 8.1) of a `ServerRequest` structure shall not be higher than the major version of the device engagement structure (see 8.2.1.1) communicated by the mdoc in the same transaction.

`token` shall contain the server retrieval token (see 8.2.1.2) which identifies the mdoc.

`docRequests` contains an array of all requested documents.

`docType` is the requested document type.

`requestInfo` may be used by the mdoc reader to provide additional information. This document does not define any key-value pairs for use in `requestInfo`. An IA infrastructure shall ignore any key-value pairs that it is not able to interpret.

`NameSpaces` contains the requested data elements and the namespace they belong to.

`DataElements` contains the requested data elements identified by their data element identifier. For each requested data element, the `IntentToRetain` variable indicates whether the mdoc verifier intends to retain the received data element. The mdoc verifier shall not retain any data, including digests and signatures, or derived data received from the mdoc, except for data elements for which the `IntentToRetain` flag was set to true in the request. To retain is defined as "to store for a period longer than necessary to conduct the transaction in realtime".

The IA infrastructure shall ignore all unknown data elements in a server retrieval mdoc request when processing the request.

An informative example of a server retrieval mdoc request can be found in D.4.2.1.1.

8.3.2.2.2.2 Server retrieval mdoc response

The server retrieval mdoc response shall be JSON encoded and shall be formatted as follows:

```
ServerResponse = {
    "version" : tstr,                   ; Version of the structure
    ? "documents" : [+JWT],             ; Returned documents
    ? "documentErrors": [+DocumentError]; For unreturned documents, optional error codes
}

JWT = tstr                              ; JWT with JWTClaimsSet as JWT Claims Set

JWTClaimsSet = {
    "doctype" : DocType,                ; Document type returned
    "namespaces" : NameSpacesResponse,
    ? "errors" : Errors,
    * tstr => any                       ; Registered claims
}

NameSpacesResponse = {
    + NameSpace => DataElementsValues   ; Returned data elements for each namespace
}

DataElementsValues = {
    + DataElementIdentifier => DataElementValue
}

DocumentError = {
    DocType => ErrorCode                ; Error codes for unreturned documents
}

Errors = {
    + NameSpace => ErrorItems               ; Error codes for each namespace
}
```

```
ErrorItems = {
    + DataElementIdentifier => ErrorCode     ; Error code per data element
}

ErrorCode = int                               ; Error code
```

`version` is the version for the `ServerResponse` structure: in the current version of this document its value shall be "1.0". If other versions are specified in the future, the major version (see 8.1) of a `ServerResponse` structure shall not be higher than the major version of the device engagement structure (see 8.2.1.1) communicated by the mdoc in the same transaction. The major version of a `ServerResponse` structure shall also not be higher than the major version of the server retrieval mdoc request (see 8.3.2.2.2.1) to which it is a response.

`documents` contains an array of all returned documents. Each document shall be returned as a JSON Web Token (JWT), as specified in RFC 7519. The claims conveyed by each JWT are in `JWTClaimsSet`. Each JWT is protected using a JSON Web Signature (JWS) as specified in 9.2.2.

`documentErrors` can contain error codes for documents that are not returned.

`doctype` is the returned document type.

`NameSpacesResponse` contains the returned mdoc data elements as part of their namespace. `DataElementsValues` contains the data element identifiers and the corresponding data element values.

`errors` can contain error codes for data elements that are not returned.

In addition to the `doctype`, `namespaces` and `errors` claims, also any registered claims (see RFC 7519) can be returned as part of `JWTClaimsSet`. Whenever any mdoc data is sent to an mdoc reader, the following registered claims shall be present in the JWT: exp (Expiration Time) and iat (Issued AT) as defined in RFC 7519.

If the server retrieval mdoc response structure does not include some data element or document requested in the server retrieval mdoc request, an error code may be returned as part of the `documentErrors` or `errors` structures. If present `ErrorCode` shall contain an error code according to Table 10. An `ErrorCode` is specific to the document or data element requested within a namespace.

Table 10 — Data handling errors

Error code	Error code message	Description
0	Data not returned	The issuing authority infrastructure does not provide the requested document or data element without any given reason. This element may be used in all cases.
Other positive integers	See description	RFU
Negative integers	See description	These error codes may be used for application-specific purposes.

An example of the server retrieval mdoc response can be found in D.4.2.1.2.

8.3.3 Data retrieval transmission technologies

8.3.3.1 Device retrieval

8.3.3.1.1 Data retrieval using Bluetooth® low energy (BLE)

8.3.3.1.1.1 General

Bluetooth®[1] low energy (BLE), implemented according to the Bluetooth Core Specification[10], may be used for device retrieval. An mdoc shall support version 4.0 of this specification. An mdoc should support version 4.2 and LE Data Packet Length Extension. An mdoc reader shall support version 4.2 and LE Data Packet Length Extension. An mdoc reader is recommended to support version 5.0 and LE 2M PHY.

Using BLE as a transmission technology consists of two phases, connection setup and data retrieval. During connection setup, the mdoc and mdoc reader connect to each other. After the connection is set up, data retrieval can be initiated.

BLE secure connections can be used if supported by both the mdoc and mdoc reader. However, an mdoc or an mdoc reader shall not require the use of BLE secure connections. Security of the transferred data is ensured by the security mechanisms specified in Clause 9.

All BLE session information shall be removed after each transaction.

As part of device engagement, an mdoc, and optionally the mdoc reader, indicates whether it supports the Central role, the Peripheral role or both, which are implemented according to the Bluetooth Core Specification[10].

NOTE It is possible for the mdoc or mdoc reader to support both roles, until data retrieval has started.

If the mdoc supports the Central role, it shall act as a GATT client. This mode is called mdoc central client mode. If the mdoc supports the Peripheral role, it shall act as a GATT server. This mode is called mdoc peripheral server mode.

If the mdoc indicates during device engagement that it supports both modes, the mdoc reader should select the mdoc central client mode.

8.3.3.1.1.2 Device engagement contents

This subclause describes the requirements for the contents of the device engagement information. Reference [13] gives further informative guidance.

Device engagement using NFC

For device engagement using NFC, the BLE alternative carrier configuration record Record Type Name shall be "application/vnd.bluetooth.le.oob". The Supplement to the Bluetooth Core Specification[11] specifies data types used for OOB data blocks. The following requirements apply to the presence of the data types in the OOB data block.

— LE Role (0x1C). This type is mandatory. It shall be used by the mdoc and mdoc reader to indicate which LE roles they support or select.

— LE Device Address (0x1B). This field is recommended. If it is available, the connection process can take a shorter time compared to using the UUID to identify the correct device to connect to.

— Complete List of 128-bit Service UUIDs (0x07). The requirements for including this field are defined below. This field shall be used to transfer the UUIDs used for connection setup, if applicable.

1) Bluetooth is the trademark of a product supplied by the Bluetooth Special Interest Group. This information is given for the convenience of users of this document and does not constitute an endorsement by ISO/IEC of the product named. Equivalent products may be used if they can be shown to lead to the same results.

NOTE When encoding the OOB data blocks, note that the Supplement to the Bluetooth Core Specification specifies that certain values should be little-endian, in Clause 1 it is stated that: "All numerical multi-byte entities and values associated with the following data types shall use little-endian byte order."

Other data types may be included in the OOB data block.

The following requirements apply for including the UUID field during NFC device engagement:

— for Negotiated Handover, if the mdoc reader supports mdoc central client mode, it shall include a UUID in the Handover Request message, to be used for mdoc central client mode;

— for Negotiated Handover, if the mdoc chooses to use mdoc peripheral server mode, it shall include a UUID in the Handover Select message, to be used for mdoc peripheral server mode;

— for Static Handover, the mdoc shall send one UUID in the handover select message, to be used for mdoc central client mode, mdoc peripheral server mode or both.

Device engagement using QR code

For device engagement using QR code, the mdoc shall indicate which modes it supports using the fields in the `BleOptions` structure as specified in 8.2.2.3. The UUIDs in the `BleOptions` structure shall be encoded using variant 1 ('10x'b) as a 16 octets element with the byte order as specified in RFC 4122, section 4.1.2. The value of the BLE Device Address field shall use the same encoding as the LE Device Address field used in device engagement using NFC as defined earlier in this subclause.

The UUID for peripheral server mode shall be present if mdoc peripheral server mode is supported and shall not be present if peripheral server mode is not supported.

The UUID for client central mode shall be present if mdoc central client mode is supported and shall not be present if central client mode is not supported.

The BLE Device Address field may be present if mdoc peripheral server mode is supported and it shall not be present if peripheral server mode is not supported. If it is available, the connection process can take a shorter time compared to using the UUID to identify the correct device to connect to.

8.3.3.1.1.3 Connection setup

The UUIDs used shall be 16-byte UUIDs that are unique for the transaction. The Peripheral device shall broadcast the service with the UUID as received or sent during device engagement in the advertising packet. The Central device is then able to scan for the UUID and connect to the advertised service. However, the Central device may use a different mechanism to identify the Peripheral device.

NOTE 1 BLE stacks in mobile devices can use scan filter and caching methods to manage congested environments and manage scan intervals for device energy consumption control. This can influence the connection time required when using UUIDs for the identification of the Peripheral device.

NOTE 2 Finding the correct device to connect to is purely a practical problem. Connecting to the wrong mdoc reader does not have security implications, since due to the security methods described in Clause 9, the mdoc and mdoc reader will not setup a session with the wrong mdoc reader. Note however, that these mechanisms do not provide complete protection against a bad actor aiming to cause a denial of service attack by advertising as a fake mdoc reader.

To ensure that the mdoc is connected to the correct mdoc reader, the mdoc may verify the Ident characteristic as described in 8.3.3.1.1.4. The Ident characteristic value shall be calculated using the following procedure:

Use HKDF as defined in RFC 5869 with the following parameters:

— Hash: SHA-256,

— IKM: `EdeviceKeyBytes` (see 9.1.1.4),

— salt: (no salt value is provided),

— info: “BLEIdent” (encoded as a UTF-8 string),

— L: 16 octets.

If the Ident characteristic received from the mdoc reader does not match the expected value, the mdoc shall terminate the connection.

NOTE 3 The purpose of the Ident characteristic is only to verify whether the mdoc is connected to the correct mdoc reader before starting data retrieval. If the mdoc is connected to the wrong mdoc reader, session establishment will fail. Connecting and disconnecting to an mdoc reader takes a relatively large amount of time and it is therefore fastest to implement methods to identify the correct mdoc reader to connect to and not to rely purely on the Ident characteristic to identify the correct mdoc reader.

After connection is setup, the GATT client may check to see if the GATT server supports the L2CAP transmission profile and, if so, use it to transfer data. See Annex A for more information. If the L2CAP transmission profile is used, 8.3.3.1.1.5, 8.3.3.1.1.6, 8.3.3.1.1.7, and 8.3.3.1.1.8 do not apply.

8.3.3.1.1.4 Service definition

Table 11 shows characteristics which the mdoc service shall contain if the mdoc is the GATT server. The services may contain other characteristics and properties besides the ones required in Table 11.

NOTE Supporting the Write property next to Write Without Response property can solve some interoperability issues. Using Write Without Response has a higher transmission rate.

Table 11 — mdoc service characteristics

Characteristic name	UUID	Mandatory properties
State	00000001-A123-48CE-896B-4C76973373E6	Notify, Write Without Response
Client2Server	00000002- A123-48CE-896B-4C76973373E6	Write Without Response
Server2Client	00000003- A123-48CE-896B-4C76973373E6	Notify

Table 12 shows characteristics which the mdoc reader service shall contain if the mdoc reader is the GATT server. The services may contain other characteristics and properties besides the ones required in Table 12.

Table 12 — mdoc reader service characteristics

Characteristic name	UUID	Mandatory properties
State	00000005- A123-48CE-896B-4C76973373E6	Notify, Write Without Response
Client2Server	00000006- A123-48CE-896B-4C76973373E6	Write Without Response
Server2Client	00000007- A123-48CE-896B-4C76973373E6	Notify
Ident	00000008- A123-48CE-896B-4C76973373E6	Read

Each service characteristic having the Notify property shall contain the Client Characteristic Configuration Descriptor, with UUID ‘0x29 0x02’ and default value of ‘0x00 0x00’. This value shall be set to ‘0x00 0x01’ by the GATT client to get notified for the characteristic associated to this descriptor.

8.3.3.1.1.5 Connection state

After the connection is setup, the GATT client shall subscribe to notifications of characteristic ‘State’ and ‘Server2Client’. For performance reasons, the GATT client should request for an MTU as high possible.

After these steps, the GATT client shall make a write without response request to 'State' where it sets the value to 0x01. This tells the GATT server that the GATT client is ready for the transmission to start.

The connection state is indicated by the 'State' characteristic. It is encoded as 1-byte binary data. Table 13 describes the different connection state values, which are communicated using Write Without Response and Notify.

Table 13 — Connection state values

Command	Data	Sender	Description
Start	0x01	GATT client	This indicates that the mdoc reader may/will begin transmission.
End	0x02	mdoc, mdoc reader	Signal to finish/terminate transaction. The mdoc reader shall use this value to signal the end of data retrieval. Both the mdoc and the mdoc reader can use this value at any time to terminate the connection. See also 9.1.1.4 for more information on session termination.

8.3.3.1.1.6 Data retrieval

Data retrieval shall start by signalling the 'Start' value to the 'State' characteristic.

The data sent shall be the `SessionEstablishment` or `SessionData` messages as defined in 9.1.1.4.

If the GATT client wants to send a message to the GATT server, it shall divide the message in parts with a length of 3 bytes less than the MTU size. It then sends these parts to the GATT server using the Write Without Response command via the 'Client2Server' characteristic. The first byte of each part is either 0x01, which indicates more messages are coming, or 0x00, to indicate it is the last part of the message.

If the GATT server wants to send a message to the GATT client, it shall divide the message in parts with a length of 3 bytes less than the MTU size. It then sends these parts to the GATT client using the Notify command via the 'Server2Client' characteristic. The first byte of each part is either 0x01, which indicates more parts are coming, or 0x00, to indicate it is the last part of the message.

The sequence of messages shall be repeated as long as necessary to finish data retrieval.

Figure 5 shows the informative sequence diagram for the data retrieval phase as described in this subclause.

Figure 5 — Data transfer sequence diagram

8.3.3.1.1.7 Connection closure

After data retrieval, the GATT client shall unsubscribe from both the 'State' and 'Server2Client' characteristics and shall disconnect from the GATT server.

8.3.3.1.1.8 Connection re-establishment

In case of a lost connection before the 'State' characteristic has been set to a value of 0x01 (e.g. the transmission has not yet started), the mdoc and mdoc reader should terminate their current BLE session and try to reconnect according to 8.3.3.1.1.3.

In case of a lost connection after the 'State' characteristic has been set to value 0x01 (e.g. the transmission of data has started), a connection shall not be re-established, and a completely new mdoc transaction shall be initiated if required.

8.3.3.1.2 Data retrieval using near field communication (NFC)

NFC may be used for device retrieval. In case NFC is used, the mdoc shall support PICC mode and the mdoc reader shall support PCD mode.

An mdoc and mdoc reader shall support short-length fields as specified in ISO/IEC 7816-4:2020, 5.2 and should support extended-length fields as specified in ISO/IEC 7816-4:2020, 5.2. An mdoc shall indicate the maximum length of command data fields and of response data fields it supports during device engagement, as specified in 8.2.2.2 for device engagement using QR code and 8.2.2.3 for device engagement using NFC. The two fields shall indicate the maximum length of the command and response data fields as defined in ISO/IEC 7816-4:2020, 5.2 supported by the mdoc.

NOTE 1 For device engagement using QR code, this information is in the NfcOptions map. For device engagement using NFC, this information is in the NFC Handover Select message.

NOTE 2 The minimum and maximum possible values for the command data field limit are 'FF' and 'FF FF', i.e. the limit is between 255 and 65 535 bytes (inclusive). The minimum and maximum possible values for the response data limit are '01 00' and '01 00 00', i.e. the limit is between 256 and 65 536 bytes (inclusive).

The mdoc reader shall respect the data field size limitations of the mdoc.

NOTE 3 The mdoc reader determines the size of both the command and the response APDUs and can, therefore, make sure that its APDU size limitations (if any) are not being violated.

An mdoc and mdoc reader shall support command chaining and response chaining as specified in ISO/IEC 7816-4:2020, 5.3.

NOTE 4 Even if extended-length APDUs are supported by the mdoc and the mdoc reader, using command or response chaining can still be necessary, because it cannot be guaranteed that an mdoc reader request or an mdoc response fits in a single APDU.

The Application IDentifier (AID) of the mdoc shall be 'A0 00 00 02 48 04 00'.

NOTE 5 The AID of the mdoc application consists of the registered application provider identifier (RID) ('A0 00 00 02 48') followed by the proprietary application identifier extension (PIX) ('04 00').

An mdoc application shall be selected using the SELECT command defined in ISO/IEC 7816-4 with the AID listed above. Table 14 and Table 15 specify the SELECT command and response APDUs.

Table 14 — SELECT command

CLA	INS	P1-P2	Lc field	Data field	Le field
'00'	'A4'	'04 0C'	'07'	'A0 00 00 02 48 04 00'	Absent

Table 15 — SELECT response

Data field	SW1 – SW2
Absent	See ISO/IEC 7816-4:2020, Table 61

After the mdoc application is selected, the mdoc reader can start data retrieval. The mdoc reader shall use the ENVELOPE command with INS = 'C3', specified in ISO/IEC 7816-4, to communicate the `SessionEstablishment` en `SessionData` messages as defined in 9.1.1.4 to the mdoc. These messages shall be encapsulated in a data object '53' as specified in ISO/IEC 7816-4:2020, 11.7.2. The mdoc shall use the ENVELOPE response to communicate the `SessionData` messages as defined in 9.1.1.4 to the mdoc reader. These messages shall be encapsulated in a data object '53' as well.

Table 16 and Table 17 specify the ENVELOPE command and response APDUs.

Table 16 — ENVELOPE command

CLA	INS	P1-P2	Lc field	Data field	Le field
'00' or '10', as defined in ISO/IEC 7816-4:2020, 5.4.1	'C3'	'0000'	Length of data field	Data object '53' or data object fragment	See below

Table 17 — ENVELOPE response

Data field	SW1 – SW2
Data object '53' or absent if an error occurred on ISO/IEC 7816-4:2020 protocol level	See ISO/IEC 7816-4:2020, Table 121, in particular '61 XY'.

For oversize incoming payload (from an mdoc reader to an mdoc), several ENVELOPE commands shall be chained as specified in ISO/IEC 7816-4:2020, 5.3. For oversize outgoing payload (from an mdoc to an mdoc reader), response chaining shall be used as specified in ISO 7816-4:2020, 5.3 and further detailed below, using one or more GET RESPONSE commands and responses.

Regarding the value of Le in the ENVELOPE and GET RESPONSE commands:

— For all ENVELOPE commands in a chain except the last one, Le shall be absent, since no data is expected in the response to these commands;

— For the last ENVELOPE command in a chain, Le shall be set to the maximum length of the response data field that is supported by both the mdoc and the mdoc reader. The mdoc reader shall encode Le as specified in ISO/IEC 7816-4:2020, 5.2;

— For the last ENVELOPE command or for a GET RESPONSE command,

  — if Le ≥ the number of available bytes, the mdoc shall include all available bytes in the response and set the status words to '90 00';

  — if Le < the number of available bytes ≤ Le + 255, the mdoc shall include as many bytes in the response as indicated by Le and shall set the status words to '61 XX', where XX is the number of available bytes remaining. The mdoc reader shall respond with a GET RESPONSE command where Le is set to XX;

  — if the number of available bytes > Le + 255, the mdoc shall include as many bytes in the response as indicated by Le and shall set the status words to '61 00'. The mdoc reader shall respond with a GET RESPONSE command where Le is set to the maximum length of the response data field that is supported by both the mdoc and the mdoc reader.

If the NFC connection is lost during data retrieval, a completely new mdoc transaction (including device engagement) shall be initiated.

8.3.3.1.3 Data retrieval using Wi-Fi Aware

8.3.3.1.3.1 General

Wi-Fi Aware may be used for device retrieval. Wi-Fi Aware shall be implemented according to the Wi-Fi Alliance Neighbor Awareness Networking Specification. The data retrieval using Wi-Fi Aware consists of three phases, connection setup, data retrieval and closure.

8.3.3.1.3.2 Connection setup

Wi-Fi Aware is setup using the information exchanged during device engagement. The Wi-Fi Alliance Neighbor Awareness Networking Specification, Version 3.1 describes the connection setup process. The Wi-Fi Alliance Neighbor Awareness Networking Specification, section 12 describes the connection setup process for NFC Negotiated Connection Handover and NFC Static Connection Handover. The structure used to transfer the required info if QR code is used for device engagement is described below.

The service name shall be calculated by the mdoc and mdoc reader for each transaction using the following mechanism.

Use HKDF as defined in RFC 5869 with the following parameters:

— Hash: SHA-256,

— IKM: `EDeviceKeyBytes` (see 9.1.1.4),

— salt: (no salt value is provided),

— info: “NANService” (encoded as a UTF-8 string),

— L: 16 octets.

The output of the HKDF shall be converted to base16 according to RFC 4648 to get the service name.

EXAMPLE “94AB45CDBDEF675162183B12AC35EFAA”.

Use of a cipher suite as defined by the Wi-Fi Alliance Neighbor Awareness Networking Specification is mandatory. The mdoc reader shall support the NCS-SK-128 and NCS-PK-2WDH-128 cipher suites, as specified by the Wi-Fi Alliance Neighbor Awareness Networking Specification. The mdoc shall support at least the NCS-SK-128 cipher suite and should support the NCS-PK-2WDH-128 cipher suite.

If NFC is used for device engagement, either the Pass-phrase Info or the DH Info shall be explicitly transferred from the mdoc to the mdoc reader during device engagement according to the Wi-Fi Alliance Neighbor Awareness Networking Specification, section 12.

NOTE 1 Since the NCS-PK-2WDH cipher suite requires both the mdoc and the mdoc reader to exchange ephemeral public keys during device engagement, it can only be used if NFC Negotiated Connection Handover is used and not if NFC Static Connection Handover or QR code is used for device engagement.

NOTE 2 The Wi-Fi Alliance Neighbor Awareness Networking Specification references RFC 8110 for the Diffie-Hellman key exchange used in the NCS-PK-2WDH cipher suites. Note that RFC 8110 mandates the support for the group nineteen curve, which is the P-256 curve. This is also the curve indicated as “256-bit random ECP group” (and having value 19) in the table referenced for the value of the D-H Key Group in Wi-Fi Alliance Neighbor Awareness Networking Specification, section 12.1.

If QR code is used for device engagement, the data used for connection setup that would be transferred as part of the Wi-Fi Aware Carrier Configuration Record if NFC were used, as defined in the Wi-Fi Alliance Neighbor Awareness Networking Specification, is transferred as part of the `WifiOptions` structure (see 8.2.2.3). Table 18 describes for each element in the `WifiOptions` structure what the corresponding field would be in the Wi-Fi Aware Carrier Configuration Record. The mdoc and mdoc reader shall comply with the requirements in Wi-Fi Alliance Neighbor Awareness Networking Specification regarding the presence and values of these fields, with the exception of the presence of the Pass-phrase field, for which the presence and associated behaviour is defined below.

Table 18 — WifiOptions elements

Key in WifiOptions	Corresponding field in Wi-Fi Carrier Configuration Record	Corresponding Sub-Field
0	Pass-phrase Info	Pass-phrase
1	Channel Info	Operating Class
2	Channel Info	Channel Number
3	Band Info	Supported Bands

Presence of the pass-phrase (i.e. the key-value pair with key = 0) in the `WifiOptions` structure is optional. When the Pass-phrase field is absent in `WifiOptions` structure, the mdoc and mdoc reader shall calculate the pass-phrase using the following mechanism.

Use HKDF as defined in RFC 5869 with the following parameters:

— Hash: SHA-256,

— IKM: `EdeviceKeyBytes` (see 9.1.1.4),

— salt: (no salt value is provided),

— info: “NANPassphrase” (encoded as a UTF-8 string),

— L: 32 octets.

The output of the HKDF calculation shall be converted using base64url-without-padding according to RFC 4648 to get the pass-phrase.

The pass-phrase field in the `WifiOptions` structure should only be present if the resulting pass-phrase contains at least 12 bytes of entropy.

During the Wi-Fi Aware service discovery procedure, the mdoc shall serve as the Service Publisher, and the mdoc reader shall serve as the Service Subscriber.

Once the Wi-Fi Aware service discovery is completed, the mdoc reader shall initiate the data path setup, and serve as the NDP Initiator; while the mdoc shall serve as the NDP Responder. The transport protocol and port number shall be transferred as part of the NDPE attribute according to Wi-Fi Alliance Neighbor Awareness Networking Specification, Version 3.1, January 2020, section 6.2.7. The mdoc and mdoc reader should use an ephemeral link-local IPv6 for each connection.

Since the IP address is transferred as part of the Wi-Fi Aware connection setup, using the IPv6 Neighbor Discovery Protocol is not necessary. The Neighbor Discovery Protocol should therefore be disabled to significantly improve connection time setup.

8.3.3.1.3.3 Data retrieval

When Wi-Fi Aware is used, mdoc data is transferred using the HTTP protocol, with the mdoc serving as the HTTP and TCP servers, and the mdoc reader serving as the HTTP and TCP client. The data retrieval shall use the HTTP POST method to transfer mdoc data. HTTP request messages shall have the following structure:

```
POST /mdoc HTTP/1.1
Host: [IPv6 address of the mdoc]
Content-Length: [content length]
Content-Type: application/cbor

[SessionEstablishment or SessionData message]
```

HTTP successful response message shall have the following structure:

```
HTTP/1.1 200 OK
Content-length: [content length]
Content-type: application/cbor

[SessionEstablishment or SessionData message]
```

The `SessionEstablishment` and `SessionData` messages are defined in 9.1.1.4.

HTTP error responses are specified in RFC 7231, section 6.1.

8.3.3.1.3.4 Closure

An mdoc or mdoc reader shall close the connection after receiving the session termination code, see 9.1.1.4.

8.3.3.2 Server retrieval

8.3.3.2.1 Data retrieval using WebAPI

The WebAPI data retrieval method may be used for server retrieval. Using this method, mdoc data is transferred using the HTTP protocol. The request method is POST. The Issuer URL (see 8.2.1.2) refers to the base server URL of an "`/identity`" WebAPI endpoint operated by the issuing authority infrastructure. The `Content-Type` header-field shall be set to "`application/json`". The host field content shall be derived from the Issuer URL element. The message body shall be the server retrieval mdoc request as defined in 8.3.2.2.2.1.

A successful response contains the HTTP status "200 OK". Table 19 defines allowed mdoc specific responses. Other HTTP responses may be returned. The `Content-Type` header shall be set to "`application/jwt`", and the "`Content-Length`" header shall be set correctly. The message body shall be the server retrieval mdoc response as defined in 8.3.2.2.2.2.

In case the issuing authority infrastructure requires interaction with the mdoc holder input to negotiate data to be shared with the mdoc reader, the issuing authority infrastructure needs to contact the mdoc. The mdoc reader shall wait for the response using HTTP long or short polling. In case of long polling, the mdoc reader should set the timeout to 120 seconds in order to avoid requests to timeout. In case of short polling the issuing authority infrastructure is sending a HTTP 202 response, including a `Retry-After` header-field with the retry delay in seconds with an empty response body. The mdoc reader shall periodically check for the response.

Table 19 — HTTP status codes

HTTP status code	HTTP status message	Description
200	OK	Successful HTTP request
202	Accepted	The HTTP request has been accepted for processing but is not yet completed.
400	Bad Request	The HTTP request was invalid or malformed.
401	Unauthorized	The provided server retrieval token was invalid.
500	Internal Server Error	The server encountered an internal server error and was not able to process the request successfully.

8.3.3.2.2 Data retrieval using OpenID Connect ('OIDC')

The OIDC data retrieval method may be used for server retrieval. If used, an mdoc reader and an issuing authority infrastructure shall implement this method as specified in this subclause.

The data retrieval process using OIDC consists of the following steps:

1. Configuration,
2. Client Registration,
3. Authorization,
4. Get ID Token,
5. Validate ID Token.

NOTE 1 It is also possible to retrieve the user claims through a userinfo endpoint (see RFC 8446), but this will be specified in a future edition of this document.

Step 1 Configuration

This step shall be used for retrieving OpenID Provider Configuration Information from the issuing authority OpenID provider as specified in OpenID Connect Discovery 1.0 incorporating errata set 1,

section 4. An issuing authority shall provide the information as a JSON document at the path formed by concatenating the string "/.well-known/openid-configuration" to the Issuer URL. Issuer URL (see 8.2.1.2) refers to the base server URL address of the issuing authority OpenID provider.

An mdoc reader shall send an OpenID Provider configuration request. The issuing authority Open ID provider shall respond with an OpenID Provider configuration response as specified in OpenID Connect Discovery 1.0 incorporating errata set 1, section 4.1 and 4.2 respectively.

Step 2 Client Registration

An mdoc reader requires a client id. If "registration_endpoint" information is available in the OpenID Provider response (see Step 1), dynamic client registration, as specified in Reference [18], may be used to obtain such a client_id. If the issuing authority infrastructure OpenID Provider does not support dynamic client registration, an mdoc reader shall obtain a client_id in another way (e.g. an out-of-band manner).

Step 3 Authorization

The mdoc reader shall use Authorization Code Flow Grant as specified in OpenID Connect Core 1.0 errata set 1, section 3.1. The client id retrieved in Step 2 shall be used in the authentication request. The server retrieval token retrieved from the mdoc (see 8.2.1.2) shall be used as an input to the “login_hint” parameter in the authentication request. The authentication request shall be sent to the authorization endpoint. Information on the authorization endpoint is included in the OpenID Provider configuration response in Step 1.

The authentication response is redirected to the mdoc according to OAuth 2.0 for Native Apps and shall be according to RFC 8252. The authentication response shall include the authorization code assigned by the authorization endpoint.

Step 4 Get ID Token

User claims shall be retrieved from the Token endpoint with an ID token. In order to retrieve user claims, an mdoc reader shall access the Token endpoint. Access information on the Token endpoint is included in the OpenID Provider configuration response.

An mdoc reader shall send a Token request containing the authorization code retrieved in Step 3. A successful Token response shall contain at least "exp" and "iat" elements as defined in OpenID Connect Core and a `docType` element as defined in 8.3.1, the `docType` element shall use the claim name “doctype”.

The following convention for naming claims shall be used for using a namespace within the OIDC framework. Each data element shall get the namespace (see 8.1) as a prefix as [`NameSpace`]:[`DataElementIdentifier`].

EXAMPLE “org.iso.18013.5.1:portrait”.

NOTE 2 OIDC currently does not support the `IntentToRetain` element specified in 8.3.2.2.2.1. The capability to use that is expected in a future edition of this document.

Step 5 Validate ID Token

An mdoc reader shall validate the ID Token according to OpenID Connect Core 1.0 errata set 1, section 3.1.3.7. The public key to verify the certificate chain is available in a JWKS (JSON Web Key Set) repository as defined in RFC 7517. The URI of the JWKS should be included in the OpenID Provider Configuration Response.

Examples of OIDC request and response messages are described in D.4.2.2.

9 Security mechanisms

9.1 Device retrieval

9.1.1 Session encryption

9.1.1.1 Purpose

Encrypting with authentication of the mdoc requests and mdoc responses with the session key protects mdoc data from eavesdropping and alteration.

9.1.1.2 Applicability

This mechanism is applicable for an mdoc using device retrieval.

9.1.1.3 Description

Session encryption uses standard ephemeral key ECDH to establish session keys for authenticated symmetric encryption.

9.1.1.4 Procedure

The following steps shall be performed as part of session encryption.

1. Device engagement. The mdoc generates a new ephemeral key pair (EDeviceKey.Priv, EDeviceKey.Pub), and includes the cipher suite identifier, the identifier of the elliptic curve to be used for key agreement and the EDeviceKey public point, as part of the device engagement structure as defined in 8.2.1.1.

2. Session establishment. The mdoc reader generates a new ephemeral key pair (EReaderKey.Priv, EReaderKey.Pub) using the elliptic curve identified by the mdoc. Session keys are derived independently by the mdoc and the mdoc reader as specified in 9.1.1.5.

   The mdoc reader encrypts the mdoc request with the appropriate session key and sends it to the mdoc together with EReaderKey.Pub in a session establishment message.

   The mdoc uses the data from the session establishment message to derive the session keys and decrypts the mdoc request.

3. Session data. The mdoc encrypts the mdoc response with the appropriate session key and sends it to the mdoc reader in a session data message.

   The mdoc reader and mdoc optionally exchange further session data messages containing additional mdoc requests and mdoc responses. If so, these requests and responses are encrypted by the mdoc reader and the mdoc using their respective session keys.

4. Session termination. The session shall be terminated if at least one of the following conditions occur.

   — After a time-out of no activity of receiving or sending session establishment or session data messages occurs. The time-out for no activity implemented by the mdoc and mdoc reader should be no less than 300 s.

   — If the mdoc does not want to receive any further requests.

   — If the mdoc reader does not want to send any further requests.

   If an mdoc or an mdoc reader does not want to send or receive any further requests, it shall initiate session termination as follows.

— If any transmission method besides BLE is used for data transmission, it shall send the status code for session termination.

— If BLE is used for data transmission, an mdoc or mdoc reader has two options to send the termination message:

  — to send the status code for session termination;

  — to send the "End" command defined in 8.3.3.1.1.5

When a session is terminated, the mdoc and mdoc reader shall perform at least the following actions:

— destruction of session keys and related ephemeral key material;

— closure of the communication channel used for data retrieval.

The session establishment message shall be CBOR encoded and formatted as follows:

```
SessionEstablishment = {
     "eReaderKey" : EReaderKeyBytes,
     "data" : bstr                    ; Encrypted mdoc request
}
```

The session data messages shall be CBOR encoded and formatted as follows:

```
SessionData = {
    ? "data" : bstr          ; Encrypted mdoc response or mdoc request
    ? "status" : uint        ; Status code
}
```

The contents of the `data` element in the session establishment and session data messages are defined in 9.1.1.5.

The mdoc and mdoc reader ephemeral keys shall be encoded as COSE_Key as defined in RFC 8152; further requirements are defined in 9.1.5.2. The structures that contain the mdoc and mdoc reader ephemeral keys shall be CBOR encoded and formatted as follows:

```
EDeviceKey = COSE_Key                          ; Containing EDeviceKey.Pub
EReaderKey = COSE_Key                          ; Containing EReaderKey.Pub
EDeviceKeyBytes = #6.24(bstr .cbor EDeviceKey)
EReaderKeyBytes = #6.24(bstr .cbor EReaderKey)
```

When cipher suite 1 (see 9.1.5.2) is used, one of the curves from Table 22 shall be used in `EDeviceKey` and `EReaderKey`. Only curves with the purpose ECDH shall be used.

The possible values of the status code are defined in Table 20. If status code 10 or 11 is returned, the `data` element shall not be present in that session data message.

Table 20 — SessionData status codes

Status code	Description	Action required
10	Error: session encryption	The session shall be terminated.
11	Error: CBOR decoding	The session shall be terminated.
20	Session termination	The session shall be terminated.

An example for session encryption can be found in D.5.1.

9.1.1.5 Cryptographic operations

The following operations shall be performed if cipher suite 1 (see 9.1.5.2) is used.

To calculate the session keys, the mdoc and the mdoc reader shall perform ECKA-DH (Elliptic Curve Key Agreement Algorithm – Diffie-Hellman) as defined in BSI TR-03111. The inputs shall be the EDeviceKey.Priv and EReaderKey.Pub for the mdoc and EReaderKey.Priv and EDeviceKey.Pub for the mdoc reader. The Z_{AB} output defined in BSI TR-03111 shall be used to derive two keys, SKReader and SKDevice.

SKReader shall be derived using HKDF as defined in RFC 5869 with the following parameters:

— Hash: SHA-256,

— IKM: Z_{AB},

— salt: SHA-256(`SessionTranscriptBytes`),

— info: “SKReader” (encoded as a UTF-8 string),

— L: 32 octets.

SKDevice shall be derived using HKDF as defined in RFC 5869 with the following parameters:

— Hash: SHA-256,

— IKM: Z_{AB},

— salt: SHA-256(`SessionTranscriptBytes`),

— info: “SKDevice” (encoded as a UTF-8 string),

— L: 32 octets.

`SessionTranscriptBytes` is defined in 9.1.5.1.

For encryption AES-256-GCM (GCM: Galois Counter Mode) as defined in NIST SP 800-38D shall be used. The mdoc reader shall encrypt its mdoc requests with SKReader, the mdoc shall encrypt its mdoc responses with SKDevice. Therefore, both the mdoc and the mdoc reader need to generate both session keys in order to be able to decrypt the messages they send and also decrypt the messages they receive.

The IV (Initialization Vector defined in NIST SP 800-38D) used for encryption shall have the default length of 12 bytes for GCM, as specified in NIST SP 800-38D. The IV shall be the concatenation of the identifier and the message counter (identifier || message counter). The identifier shall be an 8-byte value. The mdoc reader shall use the following identifier: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00. The mdoc shall use the following identifier: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x01. The mdoc and mdoc reader shall keep a separate message counter for each session key. The message counter value shall be a 4-byte big-endian unsigned integer. For the first encryption with a session key, the message counter shall be set to 1. Before each following encryption with the same key, the message counter value shall be increased by 1. A message counter value shall never be reused in any future encryption using the same key. The AAD (Additional Authenticated Data defined in NIST SP 800-38D) used as input for the GCM function shall be an empty string. The plaintext used as input for the GCM function shall be mdoc request or mdoc response. The value of the `data` element in the session establishment and session data messages as defined in 9.1.1.4 shall be the concatenation of the ciphertext and all 16 bytes of the authentication tag (ciphertext || authentication tag).

9.1.2 Issuer data authentication

9.1.2.1 Purpose

The purpose of issuer data authentication is to confirm that the mdoc data is issued by the issuing authority and that it has not changed since issuance.

9.1.2.2 Applicability

This mechanism is applicable for an mdoc supporting device retrieval.

NOTE Similar methods are described for server retrieval (see 9.2).

9.1.2.3 Description

Issuer data authentication is implemented by way of a digital signature over mdoc data, calculated by the issuing authority infrastructure using a public-private (asymmetric) key pair.

The isuing authority infrastructure calculates a message digest for each data element present on the mdoc and includes all digests in the mobile security object (MSO), defined in 9.1.2.4. The issuing authority infrastructure then digitally signs the MSO using a private key that is kept secret by and adds the digital signature to the mdoc data.

The public key belonging to the private key used for the digital signature is provided as part of a certificate. When the mdoc is presented to an mdoc reader, the mdoc reader retrieves this certificate. The mdoc reader shall then perform the inspection procedure as described in 9.3.1.

9.1.2.4 Signing method and structure for MSO

An mdoc digital signature is generated over the mobile security object (MSO). The MSO shall be CBOR encoded and formatted as follows:

```
IssuerAuth = COSE_Sign1      ; The payload is MobileSecurityObjectBytes

MobileSecurityObjectBytes = #6.24(bstr .cbor MobileSecurityObject)

MobileSecurityObject = {
    "version" : tstr,                       ; Version of the MobileSecurityObject
    "digestAlgorithm" : tstr,               ; Message digest algorithm used
    "valueDigests" : ValueDigests,          ; Digests of all data elements per namespace
    "deviceKeyInfo" : DeviceKeyInfo,
    "docType" : tstr,                       ; docType as used in Documents
    "validityInfo" : ValidityInfo
}

DeviceKeyInfo = {
    "deviceKey" : DeviceKey
    ? "keyAuthorizations" : KeyAuthorizations,
    ? "keyInfo" : KeyInfo
}

DeviceKey = COSE_Key

KeyAuthorizations = {
    ? "nameSpaces" : AuthorizedNameSpaces
    ? "dataElements" : AuthorizedDataElements
}

AuthorizedNameSpaces = [+ NameSpace]
AuthorizedDataElements = {+ NameSpace => DataElementsArray}
DataElementsArray = [+ DataElementIdentifier]

KeyInfo = { * int => any}    ; Positive integers are RFU, negative integers may be used for
proprietary use

ValueDigests = {
    + NameSpace => DigestIDs
}

DigestIDs = {
    + DigestID => Digest
}

ValidityInfo = {
    "signed" : tdate,
    "validFrom" : tdate,
    "validUntil" : tdate,
    ? "expectedUpdate" : tdate
}

DigestID = uint              ; DigestID as used in IssuerSignedItem
```

```
Digest = bstr
```

The version for the `MobileSecurityObject` structure shall be "1.0" in the current version of this document. The major version (see 8.1) shall not be higher than the major version of the mdoc response (see 8.3.2.1.2.2).

The `digestAlgorithm` and `valueDigests` are the digest algorithm identifier and the digests of the data elements as further specified in 9.1.2.5.

`deviceKeyInfo` contains the mdoc authentication public key and information related to this key. `deviceKey` contains the public part of the key pair used for mdoc authentication (see 9.1.3.4). The `deviceKey` element is encoded as an untagged `COSE_Key` element as specified in RFC 8152; further requirements are defined in 9.1.5.2.

As specified in 9.1.3.4, an mdoc can use a DeviceKey to calculate a signature or MAC over data elements as part of mdoc authentication. Within `DeviceKeyInfo`, `KeyAuthorizations` shall contain all the elements the key may sign or MAC. Authorizations can be given for a full namespace or per data element. If authorization is given for a full namespace (by including the namespace in the `AuthorizedNameSpaces` array), that namespace shall not be included in the `AuthorizedDataElements` map. If the `KeyAuthorizations` map is present, it shall not be empty.

`KeyInfo` may contain extra info about the key. Positive integers for `KeyInfo` labels are RFU. If application-specific extensions are present, they shall use negative integers for the labels.

`DigestID` is an unsigned integer that is used to match the hashes in the MSO to the data elements in the mdoc response. The Digest ID shall be unique within a namespace. To prevent the MSO leaking information on what data elements are present on a specific mdoc, there should be no correlation between the Digest ID's used for the same data element in the same namespace in different MSO's. The value shall be smaller than 2^31.

`DocType` is the document type of the document and shall be identical to the `DocType` element in the mdoc response as defined in 8.3.2.1.2.2.

The `ValidityInfo` structure contains information related to the validity of the MSO and its signature. The `signed` element is the timestamp at which the MSO signature was created. The `validFrom` element contains the timestamp before which the MSO is not yet valid. The timestamp of `validFrom` shall be equal or later than the `signed` element.

NOTE 1 A `validFrom` element with a future date can be used for when a change of mdoc data is expected in the future, for example, a change in age data elements.

The `validUntil` element contains the timestamp after which the MSO is no longer valid. The value of the timestamp shall be later than the `validFrom` element. The optional `expectedUpdate` element contains the timestamp at which the issuing authority infrastructure expects to re-sign the MSO (and potentially update data elements).

The timestamps in the `ValidityInfo` structure shall not use fractions of seconds and shall use a UTC offset of 00:00, as indicated by the character "Z".

NOTE 2 The `validUntil` element determines the validity period of the MSO and therefore, the mdoc cannot be validated after this date. mdoc data elements can provide further information on the administrative validity of the mdoc. For example, if the mdoc has an expiry date data element, this date can be later than the '`validUntil`' date of the MSO.

Since the timestamps in the `ValidityInfo` structure can provide linkability clues, the issuing authority infrastructure should set these timestamps with a precision that limits the linkability information. This can be done, for example, by setting the hh, mm and ss information to the same value on each provisioned mdoc.

The MSO is encapsulated and signed by the untagged `COSE_Sign1` structure as defined in RFC 8152 and identified as `IssuerAuth` for use in the mdoc response as defined in 8.3.2.1.2.2. Within the `COSE_Sign1`

structure, the `payload` shall be `MobileSecurityObjectBytes`. The `external_aad` field used in the `Sig_structure` shall be a bytestring of size zero.

The `alg` element (RFC 8152) shall be included as an element in the protected header. Other elements should not be present in the protected header.

The issuing authority infrastructure shall use one of the following signature algorithms for calculating the signature over the MSO: "ES256" (ECDSA with SHA-256), "ES384" (ECDSA with SHA-384), "ES512" (ECDSA with SHA-512) or "EdDSA" (EdDSA). "ES256" shall be used with curves P-256 and brainpoolP256r1. "ES384" shall be used with curves P-384, brainpoolP320r1 and brainpoolP384r1. "ES512" shall be used with curves P-521 and brainpoolP512r1. "EdDSA" shall be used with curves Ed25519 and Ed448. For verifying the signature, the mdoc reader shall support all of these signature algorithms and curves.

The recommendation in RFC 8152 on the use of deterministic ECDSA signatures does not apply to this document.

The certificate containing the public key belonging to the private key used to sign the MSO shall be included as an `x5chain` element as described in RFC: *CBOR Object Signing and Encryption (COSE): Headers for carrying and referencing X.509 certificates*. It shall be included as an unprotected header element. The `x5chain` element shall include at least one certificate and may contain more.

NOTE 3 The identifier for the `x5chain` element can be found in the IANA registry for COSE Header Parameters.

An example can be found in D.5.2.

9.1.2.5 Message digest function

The issuing authority infrastructure shall use one of the following digest algorithms: SHA-256, SHA-384 or SHA-512 as specified in ISO/IEC 10118-3. In the `digestAlgorithm` key-value pair in the MSO, the algorithms used shall be identified as defined in Table 21.

Table 21 — Digest algorithm identifiers

Digest algorithm	digestAlgorithm identifier
SHA-256	"SHA-256"
SHA-384	"SHA-384"
SHA-512	"SHA-512"

A digest shall be calculated separately for each data element present on the mdoc and stored in the MSO. The same digest algorithm shall be used for all data elements. Digests are identified by the combination of the `NameSpace` (see 8.3.1) and the `DigestID` (see 9.1.2.4). The input for the digest calculation shall be the `IssuerSignedItemBytes` element (see 8.3.2.1.2.2). Each `IssuerSignedItem` shall also contain an unpredictable random or pseudorandom value. This value shall be different for each `IssuerSignedItem` and shall have a minimum length of 16 bytes. The purpose of this value is to ensure that the digest value of the `IssuerSignedItem` by itself does not provide any information about its contents.

NOTE It is not necessary for the mdoc reader to retrieve all the data present on the mdoc to verify the received mdoc data; it can verify the signature over the whole MSO and use the Digest ID and namespace for each received data element to find and verify the digest for those data elements in the MSO.

9.1.3 mdoc authentication

9.1.3.1 Purpose

The security objective of mdoc authentication is to prevent cloning of the mdoc and to mitigate man in the middle attacks.

9.1.3.2 Applicability

This mechanism is applicable for an mdoc using device retrieval.

NOTE If a server retrieval token is retrieved by the mdoc reader in the `DeviceSignedItems` during device retrieval, mdoc authentication also ensures the authenticity of the server retrieval token. Under these conditions, therefore, mdoc authentication is also applicable for an mdoc using server retrieval.

9.1.3.3 Description

The mdoc private key, which belongs to the mdoc public key stored in the MSO, is used to authenticate the mdoc. It is also used to authenticate the response data contained in the `DeviceSignedItems` structure (see 8.3.2.1.2.2). The mdoc public key is stored in the MSO, see 9.1.2.4. The mdoc reader assumes that the mdoc is authentic only if the authentication signature or MAC is correct.

Security requirements regarding storage of credential information, including the mdoc private key are out of scope for this document. Additional information on storage of credential information can be found in Clause E.5.

NOTE Two mechanisms exist for mdoc authentication, MAC and ECDSA/EdDSA. MAC provides better privacy to the mdoc holder because it does not require the mdoc to produce a potentially non-repudiable signature over mdoc reader-provided data. The mdoc can always deny the MAC value to a third party because the mdoc reader could have produced it by itself. However, it is possible that the possibility to calculate a MAC is not available in all security environments on the mdoc.

9.1.3.4 Mechanism

The mdoc authentication key pair consists of a public and a private key (SDeviceKey.Priv, SDeviceKey.Pub). The public key is accessible through the `DeviceKey` element in the MSO. When cipher suite 1 (see 9.1.5.2) is used, one of the curves from Table 22 shall be used for the device key.

The mdoc authentication key shall be used to authenticate the mdoc in one of two ways: ECDH-agreed MAC or ECDSA / EdDSA signature. A single mdoc authentication key shall not be used to produce both MACs and signatures during its lifetime. An mdoc reader shall support both approaches.

The data that the mdoc authenticates is the `DeviceAuthenticationBytes` structure as defined below. The mdoc shall generate this structure and calculate either the MAC or signature. In order to verify the data, the mdoc reader shall generate the structure as well and validate the MAC or signature.

NOTE The `DeviceAuthenticationBytes` structure itself is not transferred as part of the mdoc response, only the resulting MAC or signature.

The device authentication structure shall be CBOR encoded and formatted as follows:

```
; For DeviceMac and DeviceSignature, use a null value for the payload.
; The detached content is DeviceAuthenticationBytes
DeviceMac = COSE_Mac0
DeviceSignature = COSE_Sign1

DeviceAuthenticationBytes = #6.24(bstr .cbor DeviceAuthentication)

DeviceAuthentication = [
    "DeviceAuthentication",
    SessionTranscript,
    DocType,                    ; Same as in mdoc response
    DeviceNameSpacesBytes       ; Same as in mdoc response
]
```

The `SessionTranscript` element is defined in 9.1.5.1.

The `DocType` and `DeviceNameSpacesBytes` shall contain the same data as in the same `document` element in the mdoc response structure (see 8.3.2.1.2.2).

An mdoc shall only authenticate response data elements in `DeviceNameSpaces` if the key it is using for mdoc authentication is authorized to authenticate these elements in the `KeyAuthorizations` structure in the MSO (see 9.1.2.4). The mdoc reader shall validate this authorization as part of validating the mdoc authentication.

If data elements are present in `DeviceNameSpaces`, an mdoc reader shall verify whether the `KeyAuthorizations` structure contains the proper authorization.

`DeviceMac` is defined in 9.1.3.5.

`DeviceSignature` is defined in 9.1.3.6.

An example can be found in D.5.3.

9.1.3.5 mdoc MAC Authentication

To authenticate the mdoc with mdoc MAC authentication, the mdoc computes the MAC of the device authentication data with an ephemeral MAC key (EMacKey) derived from the mdoc authentication private key and the mdoc reader ephemeral public key.

The following operations shall be performed when cipher suite 1 (see 9.1.5.2) is used.

To calculate the ephemeral MAC key, the mdoc and the mdoc reader shall perform ECKA-DH (Elliptic Curve Key Agreement Algorithm – Diffie-Hellman) as defined in BSI TR-03111. The inputs shall be the SDeviceKey.Priv and EReaderKey.Pub for the mdoc and EReaderKey.Priv and SDeviceKey.Pub for the mdoc reader. The Z_{AB} output defined in BSI TR-03111 shall be used to derive the ephemeral MAC key.

EMacKey shall be derived using HKDF as defined in RFC 5869 with the following parameters:

— Hash: SHA-256,

— IKM: Z_{AB},

— salt: SHA-256(`SessionTranscriptBytes`),

— info: “EMacKey” (encoded as a UTF-8 string),

— L: 32 octets.

The MAC value is contained in the tag element within `DeviceAuth` in an untagged `COSE_Mac0` structure as defined in RFC 8152 and identified as `DeviceMac`. Within the `COSE_Mac0` structure, the payload shall have a null value. The detached content is `DeviceAuthenticationBytes`. The ‘external_aad’ field shall be a bytestring of size zero.

The `alg` element (RFC 8152) shall be included as an element in the protected header. Other elements should not be present in the protected header.

RFC 8152 describes the algorithm identifiers that shall be used in the `alg` element. “HMAC 256/256” (HMAC with SHA-256) shall be used.

9.1.3.6 mdoc ECDSA / EdDSA Authentication

To authenticate the mdoc with mdoc ECDSA/EdDSA authentication, the mdoc signs the device authentication data with the mdoc authentication private key.

When cipher suite 1 is used (see 9.1.5.2) the following operations shall be performed and the mdoc shall use of the ECDSA or EdDSA curves from Table 22 for the mdoc authentication key.

The signature is contained in the signature element in an untagged COSE_Sign1 structure as defined in RFC 8152 and identified as `DeviceSignature`. Within the COSE_Sign1 structure, the payload shall have a null value. The detached content is `DeviceAuthenticationBytes`. The ‘external_aad’ fields shall be a bytestring of size zero.

The `alg` element (RFC 8152) shall be included as an element in the protected header. Other elements should not be present in the protected header. An mdoc shall use one of the following signature algorithms: “ES256” (ECDSA with SHA-256), “ES384” (ECDSA with SHA-384), “ES512” (ECDSA with SHA-512) or “EdDSA” (EdDSA). ”ES256” shall be used with curves P-256 and brainpoolP256r1. “ES384” shall be used with curves P-384, brainpoolP320r1 and brainpoolP384r1. “ES512” shall be used with curves P-521 and brainpoolP512r1. “EdDSA” shall be used with curves Ed25519 and Ed448.

The recommendation in RFC 8152 on the use of deterministic ECDSA signatures does not apply to this document.

9.1.4 mdoc reader authentication

9.1.4.1 Purpose

mdoc reader authentication uses information stored in the mdoc reader to confirm that the mdoc reader and the mdoc request are authenticated.

9.1.4.2 Applicability

This mechanism is applicable for an mdoc reader using device retrieval.

9.1.4.3 Description

A private key stored in the mdoc reader is used to authenticate the mdoc reader and to authenticate the mdoc request. The mdoc reader public key is stored in a certificate which is sent to the mdoc within the mdoc request message.

9.1.4.4 Mechanism

The mdoc reader authentication key pair consists of a public and a private key. The public key is accessible through a certificate provided with the mdoc request.

The mdoc reader authentication key may be used to authenticate the mdoc reader by ECDSA/EdDSA signature.

When cipher suite 1 is used (see 9.1.5.2) the following operations shall be performed and the mdoc reader shall use of the ECDSA or EdDSA curves from Table 22 for the mdoc reader authentication key.

The data that the mdoc reader authenticates is the `ReaderAuthentication` structure as defined below. The mdoc reader shall generate this structure and calculate the signature. In order to verify the data, the mdoc shall generate the structure as well and validate the signature.

The signature is contained in an untagged COSE_Sign1 structure as defined in RFC 8152 and identified as `ReaderAuth`. Within the COSE_Sign1 structure, the payload shall have a null value. The detached content is `ReaderAuthenticationBytes`. The ‘external_aad’ fields shall be a bytestring of size zero.

The `alg` element (RFC 8152) shall be included as an element in the protected header. An mdoc reader should use one of the following signature algorithms: “ES256” (ECDSA with SHA-256), “ES384” (ECDSA with SHA-384), “ES512” (ECDSA with SHA-512) or “EdDSA” (EdDSA). ”ES256” should be used with curves P-256 and brainpoolP256r1. “ES384” should be used with curves P-384, brainpoolP320r1 and brainpoolP384r1. “ES512” should be used with curves P-521 and brainpoolP512r1. “EdDSA” should be used with curves Ed25519 and Ed448.

The recommendation in RFC 8152 on the use of deterministic ECDSA signatures does not apply to this document.

NOTE 1 The ReaderAuthentication structure itself is not transferred as part of the mdoc request, only the resulting signature.

The reader authentication structure shall be CBOR encoded and formatted as follows:

```
; For ReaderAuth, use a null value for the payload.
; The detached content is ReaderAuthenticationBytes
ReaderAuth = COSE_Sign1

ReaderAuthenticationBytes = #6.24(bstr .cbor ReaderAuthentication)

ReaderAuthentication = [
    "ReaderAuthentication",
    SessionTranscript,
    ItemsRequestBytes           ; Same as in mdoc request
]
```

The `SessionTranscript` element is defined in 9.1.5.1.

The `ItemsRequestBytes` shall contain the same data as in the mdoc request structure (see 8.3.2.1.2.1).

`ReaderAuth` is defined above in this subclause.

The certificate containing the mdoc reader public key shall be included as a `x5chain` element as described in RFC: *CBOR Object Signing and Encryption (COSE): Headers for carrying and referencing X.509 certificates*. It shall be included as an unprotected header element. The `x5chain` element shall include at least one certificate and may contain more.

NOTE 2 The identifier for the `x5chain` element can be found in the IANA registry for COSE Header Parameters.

9.1.5 Session transcript and cipher suite

9.1.5.1 Session transcript

The session transcript structure is used in multiple security mechanisms for device retrieval. The session transcript shall be CBOR encoded and formatted as follows:

```
SessionTranscriptBytes = #6.24(bstr .cbor SessionTranscript)

SessionTranscript = [
    DeviceEngagementBytes,
    EReaderKeyBytes,
    Handover
]

DeviceEngagementBytes = #6.24(bstr .cbor DeviceEngagement)

Handover = QRHandover / NFCHandover

QRHandover= null

NFCHandover = [
    bstr        ; Binary value of the Handover Select Message
    bstr / null ; Binary value of the Handover Request Message,
                ; shall be null if NFC Static Handover was used
]
```

`DeviceEngagement` is defined in 8.2.1.1.

`EReaderKeyBytes` is defined in 9.1.1.4.

NOTE This document uses both `SessionTranscript` and `SessionTranscriptBytes` in cryptographic structures.

The content of `Handover` depends on the device engagement method that was used. If device engagement using QR code (see 8.2.2.3) was used, the contents shall be `QRHandover`. If device engagement using NFC (see 8.2.2.1) was used, the contents shall be `NFCHandover`.

The first element in the `NFCHandover` array shall be the binary value of the Handover Select Message as retrieved by the mdoc reader from the mdoc. The second element in the `NFCHandover` array shall be the

Handover Request Message sent by the mdoc reader to the mdoc during NFC Negotiated Handover, or null if the Handover Request Message was not present because NFC Static Handover was used.

9.1.5.2 Cipher suite

Device retrieval security mechanisms support multiple cipher suites to indicate which algorithms and operations shall be performed. This document only describes the algorithms and operations for one cipher suite, which is identified by the value 1. The mdoc indicates which cipher suite shall be used in the device engagement structure (see 8.2.1.1).

When cipher suite 1 is used, curves from Table 22 shall be used by the device retrieval security mechanisms. Support for all curves is mandatory for an mdoc reader.

The COSE_Key structures should not contain optional parameters except for the `x`, `y` and `crv` parameters. The mdoc and mdoc reader should ignore other optional parameters.

If the curve indicated in `EReaderKey` is a double-coordinate curve as specified in RFC 8152, section 13.1.1, the uncompressed form shall be used for the public key. For the public key in `EDeviceKey`, either the uncompressed or the compressed form may be used in case a double coordinate curve is used.

The brainpool curves shall have COSE Key Type EC2. COSE Key Types for other curves are specified in the IANA COSE registry.

Table 22 — Elliptic curves for cipher suite 1

Definition	Specification	Curve identifier	Purpose
Curve P-256	FIPS 186-4	IANA COSE registry	ECDH/ECDSA
Curve P-384	FIPS 186-4	IANA COSE registry	ECDH/ECDSA
Curve P-521	FIPS 186-4	IANA COSE registry	ECDH/ECDSA
X25519	RFC 7748	IANA COSE registry	ECDH
X448	RFC 7748	IANA COSE registry	ECDH
Ed25519	RFC 8032	IANA COSE registry	EdDSA
Ed448	RFC 8032	IANA COSE registry	EdDSA
brainpoolP256r1	RFC 5639	IANA COSE registry	ECDH/ECDSA
brainpoolP320r1	RFC 5639	IANA COSE registry	ECDH/ECDSA
brainpoolP384r1	RFC 5639	IANA COSE registry	ECDH/ECDSA
brainpoolP512r1	RFC 5639	IANA COSE registry	ECDH/ECDSA

NOTE 1 For IANA COSE registries[2)], see COSE (CBOR Object Signing and Encryption) Elliptic Curves in IANA (Internet Assigned Numbers Authority) Protocols Registries.

NOTE 2 In accordance with RFC 8152, the CDDL grammar describing a COSE_Key as used in the device retrieval security mechanisms is:

```
COSE_Key = {
    1 => int,              ; kty: key type
   -1 => int,              ; crv: EC identifier - Taken from the "COSE Elliptic Curves"
registry
   -2 => bstr,             ; x: value of x-coordinate
   ? -3 => bstr / bool     ; y: value or sign bit of y-coordinate; only applicable for EC2
key types
}
```

2) Available at https://www.iana.org/assignments/cose/cose.xhtml.

9.2 Server retrieval

9.2.1 TLS

Communication between the mdoc reader and the issuing authority infrastructure shall use Transport Layer Security with server authentication and optionally with client authentication. The mdoc reader and the issuing authority infrastructure shall support TLS version 1.2 as specified in RFC 5246 and may support TLS version 1.3 as specified in RFC 8446.

The mdoc reader shall act as the TLS client and the issuing authority infrastructure as the TLS server. The mdoc reader and the issuing authority infrastructure may support TLS client authentication. The key pairs used for the TLS authentication shall not be used for other purposes.

If the TLS server indicates in its TLS server certificate the support of the Online Certificate Status Protocol (OCSP), the TLS server shall support the TLS mechanisms to exchange the OCSP status information. More specifically for TLS version 1.2 the TLS server shall support the certificate status request for OCSP specified in RFC 6066, section 8. If the server supports TLS version 1.3 it shall support the TLS version 1.3 mechanisms to exchange the OCSP status information.

An mdoc reader requesting OCSP status information for the TLS server certificate shall use the corresponding TLS mechanisms to request this information.

A TLS version 1.2 connection shall use one of the cipher suites listed in Table 23. The mdoc reader and issuing authority infrastructure shall support TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 and TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 and should support TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256.

Table 23 — TLS v1.2 cipher suites

Cipher suite	Reference
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	RFC 8422
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	RFC 8422
TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256	RFC 7905

The key exchange shall make use of an elliptic curve listed in the NamedCurve enumeration in RFC 8422, section 5.1.1 for TLS 1.2 or RFC 8446, section 4.2.7 for TLS 1.3. No deprecated or reserved curves shall be used.

A TLS version 1.3 connection should use one of the cipher suites listed in Table 24. The mdoc reader and the issuing authority infrastructure shall support TLS_AES_128_GCM_SHA256 and should support TLS_AES_256_GCM_SHA384 and TLS_CHACHA20_POLY1305_SHA256.

Table 24 — TLS v1.3 cipher suites

Cipher suite	Reference
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	RFC 8446
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	RFC 8446
TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256	RFC 8446

9.2.2 JWS

A JWT sent by an issuing authority infrastructure to an mdoc reader shall be protected using a JSON Web Signature (JWS) as specified in RFC 7515. The JWS is signed with a private key and the JWS signer certificate corresponding to the private key shall be provided in the JWS Protected Header in the `x5c` parameter according to RFC 7515. The `alg` parameter according to RFC 7515 shall be included in the JWS Protected Header. One of the following JSON Web Algorithms (JWA; RFC 7518) shall be used:

a) ES256: ECDSA using Curve P-256 and SHA-256,

b) ES384: ECDSA using Curve P-384 and SHA-384,

c) ES512: ECDSA using Curve P-521 and SHA-512.

Critical JWS Header parameters shall not be used. Other optional JWS Header Parameters should not be used. JWS Compact Serialization shall be used.

Upon receiving a JWT/JWS, the mdoc reader shall perform the inspection procedure as described in 9.3.2.

An example can be found in D.5.4.

9.3 Validation and inspection procedures

9.3.1 Inspection procedure for issuer data authentication

The mdoc reader shall perform the following or functionally equivalent steps to verify that the received data is authentic and valid.

1. Validate the certificate included in the MSO header according to 9.3.3.
2. Verify the digital signature of the IssuerAuth structure (see 9.1.2.4) using the working_public_key, working_public_key_parameters, and working_public_key_algorithm from the certificate validation procedure of step 1.
3. Calculate the digest value for every `IssuerSignedItem` returned in the `DeviceResponse` structure according to 9.1.2.5 and verify that these calculated digests equal the corresponding digest values in the MSO.
4. Verify that the `DocType` in the MSO matches the relevant `DocType` in the `Documents` structure.
5. Validate the elements in the `ValidityInfo` structure, i.e. verify that:
   - the 'signed' date is within the validity period of the certificate in the MSO header,
   - the current timestamp shall be equal or later than the 'validFrom' element,
   - the 'validUntil' element shall be equal or later than the current timestamp.

9.3.2 Inspection procedure for JWS

The mdoc reader shall validate the received JWT, verify that the JWT is a JWS and validate the JWT according to RFC 7519, section 7.2 (or perform functionally equivalent steps) to verify that the received data is authentic and valid. In addition, the mdoc reader shall perform the following steps.

a) Validate the certificate included in the JWS header according to 9.3.3.

b) Verify that the `alg` header parameter denotes one of the JSON Web Algorithms listed in 9.2.2.

c) Verify the digital signature using the algorithm specified in the `alg` header parameter, the working_public_key, working_public_key_parameters, and working_public_key_algorithm from the JWS signer certificate validation procedure.

d) Validate the JWT claims 'iat' (issued at) and 'exp' (expiration time), i.e. verify that:

   1. The current time shall be equal or later than the 'iat' date,
   2. The 'exp' date shall be equal or later than the current time.

9.3.3 Certificate validation procedure

This subclause specifies the certification path validation procedure for certificates issued under an IACA certificate or under another CA as trust anchor. mdoc readers, mdocs and issuing authority infrastructures performing the certification path validation shall store the relevant trust anchor certificates or the information extracted from these trust anchor certificates that is required for the certification path validation in a way that preserves the availability and the authenticity of the information. Additionally, they shall have access to certificate revocation information.

This document does not mandate methods to obtain and/or to establish trust in IACA certificates and other certificates that serve as trust anchors for certification path validation. It is the responsibility of the person or organization responsible for the mdoc reader to obtain and/or to establish trust in the IACA certificates used to verify the certificates issued by the IACA, such as DS or JWS signer certificates. It is the responsibility of the issuing authority to obtain and/or to establish trust in certificates that are used as trust anchors to validate mdoc reader authentication certificates and TLS client authentication certificates.

However, examples of methods and approaches to establish such trust in IACA certificates are provided in Annex C, which describes a method for distribution for IACA certificates. In any case, the issuing authority shall publicly publish its issuing authority certificate authority (IACA) certificate. This document does not prescribe methods for the generation, administration, and safekeeping of key pairs. It is the responsibility of each issuing authority to ensure that keys are generated, administered, and protected as necessary.

mdoc readers, mdocs and issuing authority infrastructures performing certification path validation shall apply the RFC 5280, section 6.1 basic path validation. Furthermore, the following steps shall be performed for certificates issued by the IACA.

— Verify that the countryName element in the subject of the IACA certificate and the countryName element in the subject of the target certificate issued under the IACA certificate are the same.

— Verify that the stateOrProvinceName element in the subject of the IACA certificate and the stateOrProvinceName element in the subject of the target certificate issued under the IACA certificate are the same if this element is present in both certificates.

If this validation succeeds it returns the final value of the working_public_key, the working_public_key_algorithm, and the working_public_key_parameters.

Subclause B.3.1 illustrates how the certification path validation can be used for an end-entity certificate issued by the IACA.

Subclause B.3.2 illustrates how the CRL validation and revocation checking can be used for an end-entity certificate issued by the IACA.

Annex A
(informative)

BLE L2CAP transmission profile

The GATT server shall indicate support for the L2CAP transmission profile by making the L2CAP characteristic available next to the characteristics defined in 8.3.3.1.1.4. Table A.1 shows the characteristic which the GATT server shall expose. The contents of the characteristics shall be the Protocol Service Multiplexer (PSM).

Table A.1 — L2CAP service characteristics

Service	Characteristic name	UUID	Mandatory properties
mdoc	L2CAP	0000000A- A123-48CE-896B-4C76973373E6	Read
mdoc reader	L2CAP	0000000B- A123-48CE-896B-4C76973373E6	Read

If after reading the L2CAP service the GATT client decides to use the L2CAP transmission method, it shall establish an L2CAP connection-oriented channel according to the Bluetooth Core Specification, Version 5.2, December 2019, Vol 3, Part A, section 4[10].

The contents of the data sent shall be the `SessionEstablishment` or `SessionData` messages as defined in 9.1.1.4.

Annex B
(normative)

Certificate and CRL profiles

B.1 Certificate profiles

B.1.1 Overview

The certificate profiles defined in this annex are mandatory for mDLs, mDL readers and mDL issuing authority infrastructures, except for the mdoc reader authentication and TLS client certificate profiles in B.1.7 and B.1.8, which are recommended.

The IACA root certificate is the root certificate used to issue all end-entity certificates, except possibly mdoc reader authentication and TLS client authentication certificates. The allowed usage of the end-entity certificates is defined using the extended key usage extension. All end-entity certificates signed by an IACA root certificate shall contain the same country code as the IACA certificate. An IACA certificate may contain the stateOrProvinceName element to indicate that this IACA certificate only issues mDLs within a particular state or province. If the element is present in an IACA certificate, it shall also be present and have the same value in the end-entity certificates signed by that IACA certificate.

The IACA root certificate shall use the IACA root certificate profile as defined in B.1.2.

If an IACA link certificate is created, it shall use the IACA link certificate profile as defined in B.1.3.

For issuer data authentication (see 9.1.2), the issuing authority shall use the mDL document signer certificate profile as defined in B.1.4. This certificate is included in the `x5chain` element of `IssuerAuth` (see 9.1.2.4). The IACA root certificate shall not be included in the `x5chain` element.

NOTE 1 If the `x5chain` contains multiple certificates, the first element in the `x5chain` array is the mDL DS certificate. mdoc readers using only IACA root certificates for validation of DS certificates do not need to parse any element beyond the first in the `x5chain` array.

For JWS, the issuing authority shall use the JWS signer certificate profile as defined in B.1.5. This certificate is included in the `x5c` parameter of the JWS (see 9.2.2). The IACA root certificate shall not be included in the `x5c` parameter.

NOTE 2 If the `x5c` parameter contains multiple certificates, the first element in the `x5c` array is the JWS signer certificate. mdoc readers using only IACA root certificates for validation of JWS signer certificates do not need to parse any element beyond the first in the `x5c` array.

TLS server certificates (see 9.2.1) shall use the TLS server certificate profile as defined in B.1.6

For mdoc reader authentication (see 9.1.4), mDLs and mDL readers should use the mdoc reader authentication certificate profile as defined in B.1.7.

For TLS client authentication (see 9.2.1), mDL readers and issuing authority infrastructures should use the TLS client certificate profile as defined in B.1.8.

If a certificate that is issued by an IACA root certificate indicates support for OCSP, the OCSP signer certificate shall comply with the OCSP signer certificate profile as defined in B.1.9.

The CRL indicated in an IACA root certificate, an IACA link certificate and any certificate signed by an IACA root certificate shall comply with the requirements in Clause B.2.

All certificates shall be DER encoded.

For each certificate profile, the presence column indicates whether an element is mandatory (M), optional (O) or conditional (C). The criticality column indicates whether an element is critical (C) or non-critical (NC).

The following extensions shall not be used:

— PolicyMappings,

— NameConstraints,

— PolicyConstraints,

— InhibitAnyPolicy,

— FreshestCRL.

Certificates in this annex use OIDs for extended key usage extension. The OIDs have the following definition:

— id-mdl OBJECT IDENTIFIER ::= { iso(1) standard(0) 18013 5 },

— id-mdl-kp OBJECT IDENTIFIER ::= { id-mdl 1 } - - arc for extended key purposes,

— id-mdl-kp-mdlDS OBJECT IDENTIFIER ::= { id-mdl-kp 2 } - - arc for mDL DS,

— id-mdl-kp-mdlJWS OBJECT IDENTIFIER ::= { id-mdl-kp 3 } - - arc for JWS,

— id-mdl-kp-mdlReaderAuth OBJECT IDENTIFIER ::= { id-mdl-kp 6 } - - arc for mdoc reader authentication used by mDL readers,

— id-mdl-kp-mdlTLSClientAuth OBJECT IDENTIFIER ::= {id-mdl-kp 9} - - arc for TLS client authentication used by mDL readers,

— id-mdl-kp-mdlIACALink OBJECT IDENTIFIER ::= { id-mdl-kp 4 } - - arc for mDL IACA Link,

— id-mdl-kp-mdlIACA OBJECT IDENTIFIER ::= { id-mdl-kp 7} – arc for mDL IACA.

B.1.2 IACA root certificate

This certificate profile defines the IACA root certificate, establishing the allowed security parameters for interoperability. The IACA root certificate is used as the root for all certificates defined in this annex. One IACA root certificate may be used by multiple issuing authorities within one country. See Table B.1 for details.

Issuing authorities should define the validity period of the IACA root certificate as the sum of:

— the longest validity length of the end entity certificates to be issued (e.g. document signer certificates, JWS signer certificates, TLS server certificates or OCSP signer certificates);

— usage period: time during which end entity certificates will be issued;

— lead times: time required to create and disseminate the IACA root certificate before its usage period.

NOTE The longest validity length of the end entity certificates includes any document signer certificates for IDLs with a secure integrated circuit. IDLs with a secure integrated circuit and the corresponding document signer certificates typically have a longer maximum validity than document signer certificates for mDLs.

The private key usage period shall be carefully set, balancing the risk of having too many documents issued under the same IACA root certificate against the efforts and lead time required to create new IACA root certificates. The recommended period is between 3 to 5 years.

This document defines the IACA certificate as a self-signed root certificate for optimized performance and maximum cross-border interoperability.

The IACA public key can additionally be used in other PKI schemes, e.g. signed by a top root national CA or cross-signed by another authority. Those additional uses and the distribution mechanisms are outside the scope of this document.

Table B.1 — IACA root certificate

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Version	4.1.2.1	M		Shall be `v3`.
Serial number	4.1.2.2	M		Non-sequential positive, non-zero integer, shall contain at least 63 bits of output from a CSPRNG, should contain at least 71 bits of output from a CSPRNG, maximum 20 octets.
Signature	4.1.2.3	M		Value shall match the OID in the signature algorithm (below).
Issuer	4.1.2.4	M		`countryName` is mandatory. The value shall be in upper case and contain the ISO 3166-1 alpha-2 code of the issuing country, exactly the same value as in the issuing country data element. The `countryName` shall be `PrintableString`. `stateOrProvinceName` is optional. If this element is present, the element shall also be present in the end-entity certificates and hold the same value. The value shall exactly match the value of the data element "issuing_jurisdiction", if that element is present on the mDL. `organizationName` is optional. Its value is at the discretion of the IACA. `commonName` shall be present. Its value is at the discretion of the IACA. `serialNumber` is optional. If present, it shall be a `PrintableString`. Attributes that have a `DirectoryString` and for which the encoding is not listed above syntax shall be either `PrintableString` or `UTF8String`.
Validity	4.1.2.5	M		
Not before		M		Date on which the certificate validity period begins.
Not after		M		Maximum of 20 years after "Not before" date. NOTE The 20-year validity period results from the possibility of using the IACA root certificate for issuing an IDL according to ISO/IEC 18013-3, which allows the use of DS certificates with validity periods up to 15 years. If the IACA root certificate is only used to issue mDLs, a maximum validity period of 9 years is sufficient.
Subject	4.1.2.6	M		Same exact binary value as Issuer.
Subject public key info	4.1.2.7	M		
algorithm		M		`1.2.840.10045.2.1` (Elliptic curve)

Key

Presence:

M mandatory

O optional

Criticality:

C critical

NC not critical

Table B.1 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
parameters		M		Implicitly specify curve parameters through an OID associated with one of the following curves specified in FIPS 186-4: `1.2.840.10045.3.1.7` (Curve P-256) `1.3.132.0.34` (Curve P-384) `1.3.132.0.35` (Curve P-521) Or one of the following curves specified in RFC 5639: `1.3.36.3.3.2.8.1.1.7` (brainpoolP256r1) `1.3.36.3.3.2.8.1.1.9` (brainpoolP320r1) `1.3.36.3.3.2.8.1.1.11` (brainpoolP384r1) `1.3.36.3.3.2.8.1.1.13` (brainpoolP512r1)
subjectPublicKey		M		Public key shall be encoded in uncompressed form.
X.509v3 extensions	4.2	M		Further extensions may be present if they are marked non-critical.
Subject key identifier	4.2.1.2	M	NC	SHA-1 hash of the subject public key `BIT STRING` value (excluding tag, length, and number of unused bits).
Key usage	4.2.1.3	M	C	
Digital signature				0
Non-repudiation				0
Key encipherment				0
Data encipherment				0
Key agreement				0
Key certificate signature				1
CRL signature				1
Encipher only				0
Decipher only				0

Key

Presence:

M mandatory

O optional

Criticality:

C critical

NC not critical

Table B.1 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Issuer alternative name	4.2.1.7	M	NC	The issuer alternative name extension shall provide contact information for the issuer of the certificate. For that purpose, the issuer alternative name shall include at least one of — `rfc822Name`, or — `uniformResourceIdentifier`. NOTE This contact information is intended to help establish trust in the certificate and the certified key by appropriate out of band mechanisms. Note that this information is only meant for contact information and does not in itself imply any level of trust in the certificate.
Basic constraints	4.2.1.9	M	C	
CA		M		`TRUE`
pathLenConstraint		M		`0`
CRLDistributionPoints	4.2.1.13	M	NC	The 'reasons' and 'cRL Issuer' fields shall not be used.
distributionPoint		M		URI for CRL distribution point
Signature algorithm	4.1.1.2	M		Options: `1.2.840.10045.4.3.2` (ECDSA-with SHA256) `1.2.840.10045.4.3.3` (ECDSA-with SHA384) `1.2.840.10045.4.3.4` (ECDSA with SHA512)
Signature value	4.1.1.3	M		Value according to the signature algorithm. By creating this signature, the CA certifies the binding between the public key material and the subject of the certificate, i.e. the IACA.

Key
Presence:
M mandatory
O optional
Criticality:
C critical
NC not critical

B.1.3 IACA link certificate

This certificate profile defines the IACA link certificate, establishing the allowed security parameters for interoperability. The IA should generate and distribute an IACA link certificate when doing an IACA re-key. The link certificate establishes a trust path from the old IACA root certificate to the new one. See Table B.2 for details.

Table B.2 — IACA link certificate

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Version	4.1.2.1	M		Shall be `v3`.
Serial number	4.1.2.2	M		Non-sequential positive, non-zero integer, shall contain at least 63 bits of output from a CSPRNG, should contain at least 71 bits of output from a CSPRNG, maximum 20 octets.
Signature	4.1.2.3	M		Value shall match the OID in the signature algorithm (below).
Issuer	4.1.2.4	M		Same exact binary value as the subject in the old IACA root certificate, for which the respective private key is signing this link certificate.
Validity	4.1.2.5	M		
Not before		M		Date on which the link certificate validity period begins.
Not after		M		Date shall not be after the "Not After" date of the old IACA root certificate.
Subject	4.1.2.6	M		Same exact binary value as Issuer in the new IACA root certificate.
Subject public key info	4.1.2.7	M		Same as the subject public key info in the new IACA root certificate.
X.509v3 extensions	4.2	M		Further extensions may be present if they are marked non-critical.
Authority key identifier	4.2.1.1	M	NC	
keyIdentifier		M		Same value as the subject key identifier of the old IACA root certificate
Subject key identifier	4.2.1.2	M	NC	SHA-1 hash of the subject public key `BIT STRING` value (excluding tag, length, and number of unused bits).
Key usage	4.2.1.3	M	C	
Digital signature				`0`
Non-repudiation				`0`
Key encipherment				`0`
Data encipherment				`0`
Key agreement				`0`
Key certificate signature				`1`
CRL signature				`1`
Encipher only				`0`
Decipher only				`0`

Key

Presence:

M mandatory

O optional

C conditional

Criticality:

C critical

NC not critical

Table B.2 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Subject alternative name	4.2.1.6	C	NC	The presence is conditional. If the IACA wants to change the DN when doing a CA rollover, the subject alternative name extension shall include a `directoryName` entry with the new Issuer DN.
Issuer alternative name	4.2.1.7	M	NC	The issuer alternative name extension shall provide contact information for the issuer of the certificate. For that purpose, the issuer alternative name shall include at least one of — `rfc822Name`, or — `uniformResourceIdentifier`. NOTE This contact information is intended to help establish trust in the certificate and the certified key by appropriate out of band mechanisms. Note that this information is only meant for contact information and does not in itself imply any level of trust in the certificate.
Basic constraints	4.2.1.9	M	C	
CA		M		`TRUE`
pathLenConstraint		M		`0`
CRLDistributionPoints	4.2.1.13	M	NC	The 'reasons' and 'cRL Issuer' fields shall not be used.
distributionPoint		M		URI for CRL distribution point
Signature algorithm	4.1.1.2	M		Options: `1.2.840.10045.4.3.2` (ECDSA-with SHA256) `1.2.840.10045.4.3.3` (ECDSA-with SHA384) `1.2.840.10045.4.3.4` (ECDSA with SHA512)
Signature value	4.1.1.3	M		Value according to the signature algorithm. By creating this signature, the CA certifies the binding between the public key material and the subject of the certificate, i.e. the Public Key and DN of new IACA root certificate.

Key
Presence:
M mandatory
O optional
C conditional
Criticality:
C critical
NC not critical

B.1.4 Document signer certificate

This certificate is used to sign the mobile security object in the device retrieval mdoc response.

Additional information on privacy and security can be found in Clause E.6.

NOTE A method to enhance its security is for the issuing authority to use certificates that have a short lifespan (for example, a few weeks), and if a compromise is detected, to revoke the document signer certificate and reissue and disseminate to all concerned mDL updated data signed with a new document signer certificate. See Table B.3 for details.

Table B.3 — Document signer certificate

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Version	4.1.2.1	M		Shall be `v3`.
Serial number	4.1.2.2	M		Non-sequential positive, non-zero integer, shall contain at least 63 bits of output from a CSPRNG, should contain at least 71 bits of output from a CSPRNG, maximum 20 octets.
Signature	4.1.2.3	M		Value shall match the OID in the signature algorithm (below).
Issuer	4.1.2.4	M		Same exact binary value as the subject of IACA certificate.
Validity	4.1.2.5	M		
Not before		M		Date on which the certificate validity period begins.
Not after		M		Maximum of 457 days after "Not before" date
Subject	4.1.2.6	M		`countryName` is mandatory. The value shall be in upper case and contain the ISO 3166-1 alpha-2 code of the issuing country, exactly the same value as in the issuing country data element. The `countryName` shall be `PrintableString`. `stateOrProvinceName` is optional. If this element is present in the IACA root certificate, this element shall be present and hold the same value. The value shall exactly match the value of the data element "issuing_jurisdiction", if that element is present on the mDL. `organizationName` is optional. Its value is at the discretion of the IACA. `commonName` shall be present. Its value is at the discretion of the IACA. `localityName` is optional. Its value is at the discretion of the IACA. `serialNumber` is optional. If present, it shall be a `PrintableString`. Attributes that have a `DirectoryString` and for which the encoding is not listed above syntax shall be either `PrintableString` or `UTF8String`.
Subject public key info	4.1.2.7	M		

algorithm		M		If any of the curves specified below for the `parameters` field is used, the following OID must be used, as specified in RFC 5480 and RFC 5639:
				
				`1.2.840.10045.2.1` (id-ecPublicKey)
				
				For curves Ed25519 or Ed448, one of the following OIDs must be used, as specified in RFC 8410:
				
				`1.3.101.112(Curve Ed25519)`
				
				`1.3.101.113(Curve Ed448)`

Key

Presence:

M mandatory

O optional

Criticality:

C critical

NC not critical

Table B.3 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
parameters		C		This field must only be present when the `algorithm` field contains the OID `1.2.840.10045.2.1`. Implicitly specify curve parameters through an OID associated with one of the following curves specified in FIPS 186-4: `1.2.840.10045.3.1.7` (Curve P-256) `1.3.132.0.34` (Curve P-384) `1.3.132.0.35` (Curve P-521) Or one of the following curves specified in RFC 5639: `1.3.36.3.3.2.8.1.1.7` (brainpoolP256r1) `1.3.36.3.3.2.8.1.1.9` (brainpoolP320r1) `1.3.36.3.3.2.8.1.1.11` (brainpoolP384r1) `1.3.36.3.3.2.8.1.1.13` (brainpoolP512r1)
subjectPublicKey		M		For all curves except Ed25519 or Ed448, the public key shall be encoded in uncompressed form.
X.509v3 extensions	4.2	M		Further extensions may be present if they are marked non-critical.
Authority key identifier	4.2.1.1	M	NC	
keyIdentifier		M		Same value as the subject key identifier of the IACA root certificate
Subject key identifier	4.2.1.2	M	NC	SHA-1 hash of the subject public key `BIT STRING` value (excluding tag, length, and number of unused bits).
Key usage	4.2.1.3	M	C	
Digital signature				1
Non-repudiation				0
Key encipherment				0
Data encipherment				0
Key agreement				0
Key certificate signature				0
CRL signature				0
Encipher only				0
Decipher only				0
Subject alternative name	4.2.1.6	O	NC	

Key

Presence:

M mandatory

O optional

Criticality:

C critical

NC not critical

Table B.3 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Issuer alternative name	4.2.1.7	M	NC	The issuer alternative name extension shall provide contact information for the issuer of the certificate. For that purpose, the issuer alternative name shall include at least one of — `rfc822Name`, or — `uniformResourceIdentifier`. NOTE This contact information is intended to help establish trust in the certificate and the certified key by appropriate out of band mechanisms. Note that this information is only meant for contact information and does not in itself imply any level of trust in the certificate.
Extended key usage	4.2.1.12	M	C	
Key usage		M		`1.0.18013.5.1.2` (mdlDS)
CRLDistribution-Points	4.2.1.13	M	NC	The 'reasons' and 'cRLIssuer' fields shall not be used.
distributionPoint		M		URI for CRL distribution point
Signature algorithm	4.1.1.2	M		Options: `1.2.840.10045.4.3.2` (ECDSA-with SHA256) `1.2.840.10045.4.3.3` (ECDSA-with SHA384) `1.2.840.10045.4.3.4` (ECDSA with SHA512)
Signature value	4.1.1.3	M		Value according to the signature algorithm. By creating this signature, the CA certifies the binding between the public key material and the subject of the certificate, i.e. the document signer.

Key

Presence:

M mandatory

O optional

Criticality:

C critical

NC not critical

B.1.5 JWS signer certificate

The JWS signer certificate is used to sign all data returned using the server retrieval methods. See Table B.4 for details.

Table B.4 — JWS signer certificate

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Version	4.1.2.1	M		Shall be `v3`.
Serial number	4.1.2.2	M		Non-sequential positive, non-zero integer, shall contain at least 63 bits of output from a CSPRNG, should contain at least 71 bits of output from a CSPRNG, maximum 20 octets.
Signature	4.1.2.3	M		Value shall match the OID in the signature algorithm (below).
Issuer	4.1.2.4	M		Same exact binary value as the subject of IACA certificate.
Validity	4.1.2.5	M		
Not before		M		Date on which the certificate validity period begins.
Not after		M		Maximum of 457 days after "Not before" date.
Subject	4.1.2.6	M		`countryName` is mandatory. The value shall be in upper case and contain the ISO 3166-1 alpha-2 code of the issuing country, exactly the same value as in the issuing country data element. The `countryName` shall be `PrintableString`. `stateOrProvinceName` is optional. If this element is present in the IACA root certificate, this element shall be present and hold the same value. The value shall exactly match the value of the data element "issuing_jurisdiction", if that element is present on the mDL. `organizationName` is optional. Its value is at the discretion of the IACA. `commonName` shall be present. Its value is at the discretion of the IACA. `localityName` is optional. Its value is at the discretion of the IACA. `serialNumber` is optional. If present, it shall be a `PrintableString`. Attributes that have a `DirectoryString` and for which the encoding is not listed above syntax shall be either `PrintableString` or `UTF8String`.
Subject public key info	4.1.2.7	M		
algorithm		M		`1.2.840.10045.2.1` (Elliptic curve)
parameters		M		Implicitly specify curve parameters through an OID associated with a curve list in 9.2.2
subjectPublicKey		M		Public key shall be encoded in uncompressed form.
X.509v3 extensions	4.2	M		Further extensions may be present if they are marked non-critical.
Authority key identifier	4.2.1.1	M	NC	

Key

Presence:

M mandatory

O optional

Criticality:

C critical

NC not critical

Table B.4 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
keyIdentifier		M		Same value as the subject key identifier of the IACA root certificate
Subject key identifier	4.2.1.2	M	NC	SHA-1 hash of the subject public key `BIT STRING` value (excluding tag, length, and number of unused bits).
Key usage	4.2.1.3	M	C	
Digital signature				`1`
Non-repudiation				`0`
Key encipherment				`0`
Data encipherment				`0`
Key agreement				`0`
Key certificate signature				`0`
CRL signature				`0`
Encipher only				`0`
Decipher only				`0`
Subject alternative name	4.2.1.6	O	NC	
Issuer alternative name	4.2.1.7	M	NC	The issuer alternative name extension shall provide contact information for the issuer of the certificate. For that purpose, the issuer alternative name shall include at least one of — `rfc822Name`, or — `uniformResourceIdentifier`. NOTE This contact information is intended to help establish trust in the certificate and the certified key by appropriate out of band mechanisms. Note that this information is only meant for contact information and does not in itself imply any level of trust in the certificate.
Extended key usage	4.2.1.12	M	C	
Key usage		M		`1.0.18013.5.1.3` (mdlJWS)
CRLDistributionPoints	4.2.1.13	M	NC	The 'reasons' and 'cRL Issuer' fields shall not be used.
distributionPoint		M		URI for CRL distribution point

Key
Presence:
M mandatory
O optional
Criticality:
C critical
NC not critical

Table B.4 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Signature algorithm	4.1.1.2	M		Options: `1.2.840.10045.4.3.2` (ECDSA-with SHA256) `1.2.840.10045.4.3.3` (ECDSA-with SHA384) `1.2.840.10045.4.3.4` (ECDSA with SHA512)
Signature value	4.1.1.3	M		Value according to the signature algorithm. By creating this signature, the CA certifies the binding between the public key material and the subject of the certificate, i.e. the issuing authority.
Key Presence: M mandatory O optional Criticality: C critical NC not critical				

B.1.6 TLS server certificate – issuing authority

The TLS server certificate is used to protect the server retrieval methods using TLS. See Table B.5 for details.

Table B.5 — TLS server certificate: issuing authority

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Version	4.1.2.1	M		Shall be `v3`.
Serial number	4.1.2.2	M		Non-sequential positive, non-zero integer, shall contain at least 63 bits of output from a CSPRNG, should contain at least 71 bits of output from a CSPRNG, maximum 20 octets.
Signature	4.1.2.3	M		Value shall match the OID in the signature algorithm (below).
Issuer	4.1.2.4	M		Same exact binary value as the subject of IACA certificate
Validity	4.1.2.5	M		
Not before		M		Date on which the certificate validity period begins.
Not after		M		Maximum of 822 days after “Not before” date
Key Presence: M mandatory O optional C conditional Criticality: C critical NC not critical				

Table B.5 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Subject	4.1.2.6	M		`countryName` is mandatory. The value shall be in upper case and contain the ISO 3166-1 alpha-2 code of the issuing country, exactly the same value as in the issuing country data element. The `countryName` shall be `PrintableString`. `stateOrProvinceName` is optional. If this element is present in the IACA root certificate, this element shall be present and hold the same value. The value shall exactly match the value of the data element "issuing_jurisdiction", if that element is present on the mDL. `organizationName` is optional. Its value is at the discretion of the IACA. `commonName` shall be present. Its value is at the discretion of the IACA. `localityName` is optional. Its value is at the discretion of the IACA. `serialNumber` is optional. If present, it shall be a `PrintableString`. Attributes that have a `DirectoryString` and for which the encoding is not listed above syntax shall be either `PrintableString` or `UTF8String`.
Subject public key info	4.1.2.7	M		
algorithm		M		`1.2.840.10045.2.1` (Elliptic curve)
parameters		M		Implicitly specify curve parameters through an OID associated with one of the following curves specified in FIPS 186-4: `1.2.840.10045.3.1.7` (Curve P-256) `1.3.132.0.34` (Curve P-384) `1.3.132.0.35` (Curve P-521) Or one of the following curves specified in RFC 5639: `1.3.36.3.3.2.8.1.1.7` (brainpoolP256r1) `1.3.36.3.3.2.8.1.1.9` (brainpoolP320r1) `1.3.36.3.3.2.8.1.1.11` (brainpoolP384r1) `1.3.36.3.3.2.8.1.1.13` (brainpoolP512r1)
subjectPublicKey		M		Public key shall be encoded in uncompressed form.
X.509v3 extensions	4.2	M		Further extensions may be present if they are marked non-critical.

Key

Presence:

M mandatory

O optional

C conditional

Criticality:

C critical

NC not critical

Table B.5 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Authority key identifier	4.2.1.1	M	NC	
keyIdentifier		M		Same value as the subject key identifier of the IACA root certificate
Subject key identifier	4.2.1.2	M	NC	SHA-1 hash of the subject public key `BIT STRING` value (excluding tag, length, and number of unused bits).
Key usage	4.2.1.3	M	C	
Digital signature				`1` (mandatory)
Non-repudiation				`0`
Key encipherment				`0`
Data encipherment				`0`
Key agreement				`0`
Key certificate signature				`0`
CRL signature				`0`
Encipher only				`0`
Decipher only				`0`
Subject alternative name	4.2.1.6	M	NC	
dNSName		M		Internet domain name of the server. Can have more than one dNSName.
Issuer alternative name	4.2.1.7	M	NC	The issuer alternative name extension shall provide contact information for the issuer of the certificate. For that purpose, the issuer alternative name shall include at least one of — `rfc822Name`, or — `uniformResourceIdentifier`. NOTE This contact information is intended to help establish trust in the certificate and the certified key by appropriate out of band mechanisms. Note that this information is only meant for contact information and does not in itself imply any level of trust in the certificate.
Extended key usage	4.2.1.12	M	C	
id-kp-serverAuth		M		TLS server authentication
id-kp-clientAuth		O		TLS client authentication. Optional, to handle particular cases where the issuing authority service may need to act also as TLS client of third-party systems.

Key

Presence:

M mandatory

O optional

C conditional

Criticality:

C critical

NC not critical

Table B.5 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
CRLDistributionPoints	4.2.1.13	M	NC	The 'reasons' and 'cRL Issuer' fields shall not be used.
distributionPoint		M		URI for CRL distribution point
Private internet extensions				
Authority information access	4.2.2.1	C	NC	Conditional, shall be present if the IACA has an OCSP service.
Access description **OCSP**		C		Conditional, shall be present if the CA issuing this certificate has an OCSP service.
accessMethod		M		1.3.6.1.5.5.7.48.1 (OCSP)
accessLocation		M		URI for corresponding OCSP service
Signature algorithm	4.1.1.2	M		Options: 1.2.840.10045.4.3.2 (ECDSA-with SHA256) 1.2.840.10045.4.3.3 (ECDSA-with SHA384) 1.2.840.10045.4.3.4 (ECDSA with SHA512)
Signature value	4.1.1.3	M		Value according to the signature algorithm. By creating this signature, the CA certifies the binding between the public key material and the subject of the certificate, i.e. the IACA online service.
Key Presence: M mandatory O optional C conditional Criticality: C critical NC not critical				

B.1.7 mdoc reader authentication

The mDL reader should use the certificate profile according to Table B.6 for mdoc reader authentication (see 9.1.4).

Table B.6 — mdoc reader authentication

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Version	4.1.2.1	M		Shall be `v3`.
Serial number	4.1.2.2	M		Non-sequential positive, non-zero integer, shall contain at least 63 bits of output from a CSPRNG, should contain at least 71 bits of output from a CSPRNG, maximum 20 octets.
Signature	4.1.2.3	M		Value shall match the OID in the signature algorithm (below).
Issuer	4.1.2.4	M		The same binary value as the Subject of a CA certificate used for mdoc reader authentication. NOTE 1 This CA certificate, and the manner in which it is trusted by an mDL, is outside the scope of this document.
Validity	4.1.2.5	M		
Not before		M		Date on which the certificate validity period begins.
Not after		M		Maximum of 1 187 days after "Not before" date
Subject	4.1.2.6	M		`commonName` shall be present. Other elements may be present in the Subject field.
Subject public key info	4.1.2.7	M		

algorithm		M		If any of the curves specified below for the `parameters` field is used, the following OID must be used, as specified in RFC 5480 and RFC 5639:
				`1.2.840.10045.2.1` (id-ecPublicKey)
				For curves Ed25519 or Ed448, one of the following OIDs must be used, as specified in RFC 8410:
				`1.3.101.112(Curve Ed25519)`
				`1.3.101.113(Curve Ed448)`

Key

Presence:

M mandatory

O optional

C conditional

Criticality:

C critical

NC not critical

Table B.6 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
parameters		C		This field must only be present when the `algorithm` field contains the OID `1.2.840.10045.2.1`. Implicitly specify curve parameters through an OID associated with one of the following curves specified in FIPS 186-4: `1.2.840.10045.3.1.7` (Curve P-256) `1.3.132.0.34` (Curve P-384) `1.3.132.0.35` (Curve P-521) Or one of the following curves specified in RFC 5639: `1.3.36.3.3.2.8.1.1.7` (brainpoolP256r1) `1.3.36.3.3.2.8.1.1.9` (brainpoolP320r1) `1.3.36.3.3.2.8.1.1.11` (brainpoolP384r1) `1.3.36.3.3.2.8.1.1.13` (brainpoolP512r1)
subjectPublicKey		M		For all curves except Ed25519 or Ed448, the public key shall be encoded in uncompressed form.
X.509v3 extensions	4.2	M		Further extensions may be present if they are marked non-critical.
Authority key identifier	4.2.1.1	M	NC	
keyIdentifier		M		Same value as the subject key identifier of the Issuer CA certificate
Subject key identifier	4.2.1.2	M	NC	SHA-1 hash of the subjectPublicKey `BIT STRING` value (excluding tag, length, and number of unused bits).
Key usage	4.2.1.3	M	C	
Digital signature				`1` (mandatory)
Non-repudiation				`0`
Key encipherment				`0`
Data encipherment				`0`
Key agreement				`0`
Key certificate signature				`0`
CRL signature				`0`
Encipher only				`0`
Decipher only				`0`

Key

Presence:

M mandatory

O optional

C conditional

Criticality:

C critical

NC not critical

Table B.6 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Issuer alternative name	4.2.1.7	C	NC	Conditional, this extension shall be present if the certificate is issued by an IACA. The issuer alternative name extension shall provide contact information for the issuer of the certificate. For that purpose, the issuer alternative name shall include at least one of — `rfc822Name`, or — `uniformResourceIdentifier`. NOTE 2 This contact information is intended to help establish trust in the certificate and the certified key by appropriate out of band mechanisms. Note that this information is only meant for contact information and does not in itself imply any level of trust in the certificate.
Extended key usage	4.2.1.12	M	C	
		M		`1.0.18013.5.1.6` (mdlReaderAuth)
CRLDistribution-Points	4.2.1.13	M	NC	The 'reasons' and 'cRL Issuer' fields shall not be used.
distributionPoint		M		URI for CRL distribution point
Private internet extensions				
Authority information access	4.2.2.1	C	NC	Conditional, shall be present if the CA issuing this certificate has an OCSP service or would like to indicate other Access Description elements.
Access description **OCSP**		C		Conditional, shall be present if the CA issuing this certificate has an OCSP service.
accessMethod		M		`1.3.6.1.5.5.7.48.1` (OCSP)
accessLocation		M		URI for corresponding OCSP service
Signature algorithm	4.1.1.2	M		Options: `1.2.840.10045.4.3.2` (ECDSA-with SHA256) `1.2.840.10045.4.3.3` (ECDSA-with SHA384) `1.2.840.10045.4.3.4` (ECDSA with SHA512)
Signature value	4.1.1.3	M		Value according to the signature algorithm. By creating this signature, the CA certifies the binding between the public key material and the subject of the certificate, i.e. the mDL reader.

Key
Presence:
M mandatory
O optional
C conditional
Criticality:
C critical
NC not critical

B.1.8 TLS client authentication certificate

The mdoc reader should use the certificate profile according to Table B.7 for TLS client authentication (see 9.2.1).

Table B.7 — TLS client authentication certificate

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Version	4.1.2.1	M		Shall be `v3`.
Serial number	4.1.2.2	M		Non-sequential positive, non-zero integer, shall contain at least 63 bits of output from a CSPRNG, should contain at least 71 bits of output from a CSPRNG, maximum 20 octets.
Signature	4.1.2.3	M		Value shall match the OID in the signature algorithm (below).
Issuer	4.1.2.4	M		The same binary value as the Subject of a CA certificate used for TLS client authentication. NOTE 1 This CA certificate, and the manner in which it is trusted by an mDL, is outside the scope of this document.
Validity	4.1.2.5	M		
Not before		M		Date on which the certificate validity period begins.
Not after		M		Maximum of 1 187 days after "Not before" date
Subject	4.1.2.6	M		`commonName` shall be present. Other elements may be present in the Subject field.
Subject public key info	4.1.2.7	M		
algorithm		M		`1.2.840.10045.2.1` (Elliptic curve)
parameters		M		Implicitly specify curve parameters through an OID associated with one of the following curves specified in FIPS 186-4: `1.2.840.10045.3.1.7` (Curve P-256) `1.3.132.0.34` (Curve P-384) `1.3.132.0.35` (Curve P-521) Or one of the following curves specified in RFC 5639: `1.3.36.3.3.2.8.1.1.7` (brainpoolP256r1) `1.3.36.3.3.2.8.1.1.9` (brainpoolP320r1) `1.3.36.3.3.2.8.1.1.11` (brainpoolP384r1) `1.3.36.3.3.2.8.1.1.13` (brainpoolP512r1)
subjectPublicKey		M		Public key shall be encoded in uncompressed form.
X.509v3 extensions	4.2	M		Further extensions may be present if they are marked non-critical.

Key

Presence:

M mandatory

O optional

C conditional

Criticality:

C critical

NC not critical

Table B.7 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Authority key identifier	4.2.1.1	M	NC	
keyIdentifier		M		Same value as the subject key identifier of the Issuer CA certificate
Subject key identifier	4.2.1.2	M	NC	SHA-1 hash of the subjectPublicKey `BIT STRING` value (excluding tag, length, and number of unused bits).
Key usage	4.2.1.3	M	C	
Digital signature				`1` (mandatory)
Non-repudiation				`0`
Key encipherment				`0`
Data encipherment				`0`
Key agreement				`0`
Key certificate signature				`0`
CRL signature				`0`
Encipher only				`0`
Decipher only				`0`
Issuer alternative name	4.2.1.7	C	NC	Conditional, this extension shall be present if the certificate is issued by an IACA. The issuer alternative name extension shall provide contact information for the issuer of the certificate. For that purpose, the issuer alternative name shall include at least one of — `rfc822Name`, or — `uniformResourceIdentifier`. NOTE 2 This contact information is intended to help establish trust in the certificate and the certified key by appropriate out of band mechanisms. Note that this information is only meant for contact information and does not in itself imply any level of trust in the certificate.
Extended key usage	4.2.1.12	M	C	
		M		`1.0.18013.5.1.9` (mdlTLSClientAuth)
id-kp-clientAuth		M		Allows the usage of TLS based client authentication.
CRLDistributionPoints	4.2.1.13	M	NC	The 'reasons' and 'cRL Issuer' fields shall not be used.
distributionPoint		M		URI for CRL distribution point

Key

Presence:

M mandatory

O optional

C conditional

Criticality:

C critical

NC not critical

Table B.7 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Private internet extensions				
Authority information access	4.2.2.1	C	NC	Conditional, shall be present if the CA issuing this certificate has an OCSP service or would like to indicate other Access Description elements.
Access description **OCSP**		C		Conditional, shall be present if the CA issuing this certificate has an OCSP service.
accessMethod		M		`1.3.6.1.5.5.7.48.1` (OCSP)
accessLocation		M		URI for corresponding OCSP service
Signature algorithm	4.1.1.2	M		Options: `1.2.840.10045.4.3.2` (ECDSA-with SHA256) `1.2.840.10045.4.3.3` (ECDSA-with SHA384) `1.2.840.10045.4.3.4` (ECDSA with SHA512)
Signature value	4.1.1.3	M		Value according to the signature algorithm. By creating this signature, the CA certifies the binding between the public key material and the subject of the certificate, i.e. the mDL reader.
Key Presence: M mandatory O optional C conditional Criticality: C critical NC not critical				

B.1.9 OCSP signer certificate

The OCSP signer certificate is used to sign OCSP messages. See Table B.8 for details.

Table B.8 — OCSP signer certificate

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Version	4.1.2.1	M		Shall be `v3`.
Serial number	4.1.2.2	M		Non-sequential positive, non-zero integer, shall contain at least 63 bits of output from a CSPRNG, should contain at least 71 bits of output from a CSPRNG, maximum 20 octets.
Signature	4.1.2.3	M		Value shall match the OID in the signature algorithm (below).
Issuer	4.1.2.4	M		Same exact binary value as the subject of IACA certificate
Validity	4.1.2.5	M		
Not before		M		Date on which the certificate validity period begins.
Not after		M		If the OCSP signer certificate supports the CRLDistributionPoints extension: Maximum of 457 days after "Not before" date. If the OCSP signer certificate supports the Revocation Checking of an Authorized Responder extension: Maximum of 90 days after "Not before" date.
Subject	4.1.2.6	M		`countryName` is mandatory. The value shall be in upper case and contain the ISO 3166-1 alpha-2 code of the issuing country, exactly the same value as in the issuing country data element. The `countryName` shall be `PrintableString`. `stateOrProvinceName` is optional. If this element is present in the IACA root certificate, this element shall be present and hold the same value. The value shall exactly match the value of the data element "issuing_jurisdiction", if that element is present on the mDL. `organizationName` is optional. Its value is at the discretion of the IACA. `commonName` shall be present. Its value is at the discretion of the IACA. `localityName` is optional. Its value is at the discretion of the IACA. `serialNumber` is optional. If present, it shall be a `PrintableString`. Attributes that have a `DirectoryString` and for which the encoding is not listed above syntax shall be either `PrintableString` or `UTF8String`.
Subject public key info	4.1.2.7	M		
algorithm		M		`1.2.840.10045.2.1` (Elliptic curve)

Key

Presence:

M mandatory

O optional

C conditional

Criticality:

C critical

NC not critical

Table B.8 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
parameters		M		Implicitly specify curve parameters through an OID associated with one of the following curves specified in FIPS 186-4: `1.2.840.10045.3.1.7` (Curve P-256) `1.3.132.0.34` (Curve P-384) `1.3.132.0.35` (Curve P-521) Or one of the following curves specified in RFC 5639: `1.3.36.3.3.2.8.1.1.7` (brainpoolP256r1) `1.3.36.3.3.2.8.1.1.9` (brainpoolP320r1) `1.3.36.3.3.2.8.1.1.11` (brainpoolP384r1) `1.3.36.3.3.2.8.1.1.13` (brainpoolP512r1)
subjectPublicKey		M		Public key shall be encoded in uncompressed form.
X.509v3 extensions	4.2	M		Further extensions may be present if they are marked non-critical.
Authority key identifier	4.2.1.1	M	NC	
keyIdentifier		M		Same value as the subject key identifier of the IACA root certificate
Subject key identifier	4.2.1.2	M	NC	SHA-1 hash of the subject public key `BIT STRING` value (excluding tag, length, and number of unused bits).
Key usage	4.2.1.3	M	C	
Digital signature				1
Non-repudiation				0
Key encipherment				0
Data encipherment				0
Key agreement				0
Key certificate signature				0
CRL signature				0
Encipher only				0
Decipher only				0
Subject alternative name	4.2.1.6	O	NC	

Key

Presence:

M mandatory

O optional

C conditional

Criticality:

C critical

NC not critical

Table B.8 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Issuer alternative name	4.2.1.7	M	NC	The issuer alternative name extension shall provide contact information for the issuer of the certificate. For that purpose, the issuer alternative name shall include at least one of — `rfc822Name`, or — `uniformResourceIdentifier`. NOTE This contact information is intended to help establish trust in the certificate and the certified key by appropriate out of band mechanisms. Note that this information is only meant for contact information and does not in itself imply any level of trust in the certificate.
Extended key usage	4.2.1.12	M	C	
id-kp-OCSPSigning		M		OCSP signing delegation, see RFC 6960.
CRLDistribution-Points	4.2.1.13	C	NC	Either this extension or the Revocation checking of an authorized responder extension shall be present. The 'reasons' and 'cRL Issuer' fields shall not be used.
distributionPoint		M		URI for CRL distribution point
Revocation checking of an authorized responder		C	NC	See RFC 6960, 4.2.2.2.1. Either this extension or the CRLDistributionPoints extension shall be present.
id-pkix-ocsp-nocheck				
Signature algorithm	4.1.1.2	M		Options: `1.2.840.10045.4.3.2` (ECDSA-with SHA256) `1.2.840.10045.4.3.3` (ECDSA-with SHA384) `1.2.840.10045.4.3.4` (ECDSA with SHA512)
Signature value	4.1.1.3	M		Value according to the signature algorithm. By creating this signature, the CA certifies the binding between the public key material and the subject of the certificate.

Key
Presence:
M mandatory
O optional
C conditional
Criticality:
C critical
NC not critical

B.2 CRL profile

An IACA shall generate certificate revocation information in accordance with the Certificate Revocation List (CRL) format specified in Table B.9. This CRL shall be a full and complete CRL, i.e. list all unexpired certificates issued by the IACA that have been revoked for any reason. This CRL may contain revocation information for IACA issued certificates that are not standardized in this document. An IACA shall not use an indirect or delta CRL.

If no certificates have been revoked since the last CRL was issued, an IACA shall issue a new CRL at least every 90 days. An IACA may issue CRLs more frequently than every 90 days.

If a certificate is revoked, the IACA shall issue a new CRL indicating this revocation within 48 hours.

The CRL profile as defined in Table B.9 shall be used.

Table B.9 — CRL profile

CRL component	Section in RFC 5280	Presence	Criticality	Comments
Version	5.1.2.1	M		Shall be `v2`.
Signature	5.1.2.2	M		Value shall match the OID in the signature algorithm (below).
Issuer name	5.1.2.3	M		Same exact binary value as Subject of IACA certificate
This update	5.1.2.4	M		Issue date of this CRL
Next update	5.1.2.5	M		The next CRL will be issued no later than the next update date.
Revoked certificates	5.1.2.6	C		Conditional, shall not be present if there are no revoked certificates. If present, shall not be empty. Each CRL entry in the revoked certificates list shall contain the serial number of the revoked certificate and the revocation date. CRL entry extensions shall not be used.
CRL extensions	5.2			Further extensions shall not be present.
Authority key identifier	5.2.1	M	NC	
keyIdentifier		M		Same value as the subject key identifier of the IACA certificate.
CRL number	5.2.3	M	NC	Sequential CRL number, increased monotonically at each new CRL issued.
Signature algorithm	5.1.1.1	M		Options: `1.2.840.10045.4.3.2` (ECDSA-with SHA256) `1.2.840.10045.4.3.3` (ECDSA-with SHA384) `1.2.840.10045.4.3.4` (ECDSA with SHA512)
Signature value	5.1.1.2	M		Value according to signature algorithm.

Key
Presence:
M mandatory
C conditional
Criticality:
C critical
NC not critical

B.3 Certificate path and CRL validation examples

B.3.1 Certificate path validation example

This subclause gives an example on how the requirements from 9.3.3 apply to the certification path validation of an end-entity certificate issued by the IACA. This example does not include all steps from the path validation procedure as defined in RFC 5280. Those steps can be included when appropriate.

Use the following items from RFC 5280, section 6.1.1 as input values:

a) path length is 0;

b) current date/time;

d) Take from the IACA certificate:

1. issuer name,
2. public key algorithm,
3. public key,
4. public key parameters associated with the public key.

Initialize the following values from RFC 5280, section 6.1.2:

g) working_public_key_algorithm from IACA certificate;

h) working_public_key from IACA certificate;

i) working_public_key_parameters from IACA certificate;

j) working_issuer_name from IACA certificate;

k) Path length is 0.

Perform the following steps from RFC 5280, section 6.1.3:

a) verify step 1), 2), 3) and 4). If step 3) determines the revocation status by means of a CRL, B.3.2 gives an example on how the CRL validation can be performed.

The following steps are to be performed.

— Assign the certificate subjectPublicKey to working_public_key.

— Assign the subjectPublicKeyInfo parameters, i.e. the namedCurve, to the working_public_key_parameters variable.

— Assign the certificate subjectPublicKey algorithm to the working_public_key_algorithm variable.

— Verify the extended key usage extension in the target certificate contains the identifier for the certificate type as specified in Annex B.

— Process all critical extensions present in the certificate. Reject the certificate if it contains a critical extension that is not recognized or that contains information that cannot be processed.

The following steps are to be performed.

— Process any other recognized non-critical extensions present in the certificate. Reject the certificate if these extensions contain information that cannot be processed.

— Verify that the countryName element in the subject of the IACA certificate and countryName element in the target certificate are the same.

B.3.2 CRL validation example

This subclause gives an example on how the requirements from 9.3.3 apply to the CRL validation and certificate revocation checking of an end-entity certificate issued by the IACA.

Use the following items from RFC 5280, section 6.3.1 as input values:

a) certificate: certificate serial number and issuer name.

Initialize the following values from RFC 5280, section 6.3.2:

b) initialize the cert_status to the special value UNREVOKED.

Perform the following steps from RFC 5280, section 6.3.3:

a) obtain the latest available CRL of the IACA that issued the certificate. The latest available CRL may be a cached list, e.g. on the mDL reader;

b) verify that the CRL and the target certificate are issued by the same IACA;

g) validate the signature on the CRL using the public key from the IACA certificate;

j) search for the target certificate on the CRL. If an entry is found that matches the certificate issuer and serial number set the cert_status variable to UNSPECIFIED. If no entry is found, the cert_status variable stays UNREVOKED.

If step a), b) or g) is not successful, set the cert_status variable to UNDETERMINED and terminate the process.

Annex C
(informative)

Verified issuer certificate authority list (VICAL) provider

C.1 mDL VICAL provider policy and security requirements

C.1.1 General

C.1.1.1 Overview

The decentralized PKI trust model adopted by the mDL requires a mechanism to distribute and disseminate the set of certification authorities' certificates by issuing authorities. Furthermore, the lack of a global organization having oversight over the mDL ecosystem and willing to play an operational role (as is the case of ICAO for the electronic passport) limits the possibility of having a single central repository with all the IACA certificates and working as the reference trust anchor for all mDL participants.

In this context, a mechanism referred to as VICAL is hereby described whereby an entity (Provider) can compile, operate and provide such a trust anchor in the form of a service to mDL participants. As this service plays a critical role on the overall security and interoperability of the mDLs, a minimum set of security requirements are defined.

The VICAL Provider shall document the service it provides in a policy of technical and procedural controls. The policy shall follow the structure of this annex and be compliant with the respective requirements, which are mostly based on ISO/IEC 15408 and ISO/IEC 19790. VICAL providers can further extend the policy with additional matters (e.g. business, legal, etc.). This policy should not be the only item used to assess the trustfulness of a VICAL Provider.

This annex does not prescribe the nature or governance of a VICAL Provider. In particular, it does not preclude a scenario where multiple VICAL Providers may coexist, from public and/or private entities, competing and/or collaborating. However, it is expected that the VICAL information provided is consistent amongst different VICAL Providers.

Finally, the VICAL is provided as one possible mechanism of setting a secure and interoperable Trust Model. It does not preclude other possible mechanisms, such as bilateral and/or regional agreements.

This annex may be used for mDL as well as IDL as specified in ISO/IEC 18013-1, ISO/IEC 18013-2 and ISO/IEC 18013-3.

C.1.1.2 Document name and identification

This Policy is identified and can be referred to through the following OID:

```
id-idl-ml-policy OBJECT IDENTIFIER ::= {
    iso(1) standard(0) driving-licence (18013) part-5(5)
    VICAL(3) 1}
```

Policies of VICAL providers shall be uniquely identifiable, including all published revisions.

C.1.1.3 VICAL participants

This subclause provides an overview of the VICAL participants in the mDL.

C.1.1.3.1 VICAL providers

VICAL providers are organizations responsible for delivering the VICAL, including activities such as:

— finding and regularly validating issuing authorities' point of contacts;

— collecting information from issuing authorities through the respective point of contacts;

— compiling the VICAL;

— creating and securing the VICAL according to the defined format;

— distributing the VICAL amongst subscribers;

— updating the VICAL regularly.

C.1.1.3.2 Issuing authorities

For the definition of issuing authority, refer to Clause 3.

C.1.1.3.3 Issuing authorities' point of contact

Each issuing authority shall designate a point of contact, i.e. the unique endpoint for the trusted communication channel handling all communication between the VICAL Provider and the issuing authority. The point of contact may be the same as the initial contact point (see C.1.3.2).

The point of contact can have multiple formats, including but not limited to a person, P.O. box, an email address, dedicated telephone line, telefax, etc. See C.1.3.2 for limitations on the initial point of contact.

C.1.1.3.4 Subscribers

Subscribers receive the VICAL from the VICAL Provider and may redistribute it amongst relying parties, according to the licensing terms of the VICAL, if any.

A subscriber can simultaneously play the role of a relying party and vice-versa.

C.1.1.3.5 Relying parties

Relying parties make use of the received VICAL to validate the mDLs presented for validation.

C.1.1.4 VICAL usage

The VICAL contains the CA certificates intended to be used by relying parties as the trust anchor for the verification of authenticity and integrity of mDLs.

In the absence of VICALs that include all existing CA certificates from all Issuing Authorities it may happen that relying parties may have to use and combine several VICALs to increase coverage.

C.1.1.5 Policy administration

The VICAL Provider shall publish under this subclause the official contacts of the person or department responsible for the Policy.

The contacts shall include name, mailing address, telephone number, and email address as minimum information.

C.1.2 Publication and repository responsibilities

The VICAL Provider shall make the VICAL available to the subscribers in a secure, mutually trusted and authenticated channel, with periodic updates. The VICAL provider should make updates available at

least every 90 days. The VICAL provider should also establish procedures for extraordinary updates for emergency cases.

C.1.3 Identification and authentication

C.1.3.1 Naming

Issuing Authorities are uniquely identified by the 2-letter ISO 3166-1 country code. In the special cases of multiple different Issuing Authorities within a country, the stateOrProvinceName / issuing jurisdiction (See Annex B and 7.2.1) can be used.

The naming of certification authorities used by an issuing authority shall follow the guidelines of the certificate profiles defined and the format established in VICAL syntax notation.

C.1.3.2 Initial identity validation

The initial identity validation of the issuing authority is a critical step on the overall security of the VICAL, as it is the first step to establishing a trusted communication channel between the VICAL provider and the issuing authorities.

After a trusted communication channel is established, the VICAL provider and the issuing authority can securely and reliably exchange information to be published in the VICAL.

As a minimum, the VICAL provider shall undertake the following steps.

a) Assess the legitimacy of and identify an initial contact point for the entity claiming to be an issuing authority. This can be achieved by a number of ways, including checking:

   — against a governmental authoritative source,

   — a regional/continental representative association (or similar body) of Issuing Authorities,

   — official recognized directorate of Issuing Authorities,

   — international conventions identifying Issuing Authorities,

   — official reference by another issuing authority previously trusted,

   — other methods that clearly and undoubtedly identify an issuing authority as legitimate.

b) Initiate contact with the issuing authority's initial contact point. Use this initial contact point to engage with the issuing authority in setting up a trusted communication channel and the protocol for its use. Sensitive material used to set up the trusted communication channel shall be exchanged using out-of-band communication. The protocol shall guarantee the integrity, authenticity, non-repudiation and confidentiality of the information to be exchanged.

NOTE An IACA link certificate is an example of a trusted communication channel.

After the trusted communication channel is established, the Initial Identity Validation is concluded and the parties can then start exchanging information required for the VICAL.

The VICAL provider shall regularly check that each assigned point of contact is still valid and the established communication channel provides the security guarantees, and perform the necessary updates if any required.

C.1.3.3 Identification and authentication for re-key requests

In the context of the VICAL, a re-key request is understood as the process to add a new CA to the records of the issuing authority.

This process shall use the trusted communication channel established at the initial identity validation (see C.1.3.2). If the trusted communication channel is not considered secure (for example, weak keys or algorithm in use, lost keys, key renewal, etc.), a new trusted communication channel shall be established following the same requirements.

C.1.3.4 Identification and authentication of suspension requests

Suspension requests from issuing authorities shall be communicated through the trusted communication channels established at the initial identity validation process (see C.1.3.2).

The VICAL Provider is also reserved the right to proceed unilaterally with the suspension of a CA on the grounds of a documented process and criteria.

C.1.4 VICAL life-cycle operational requirements

C.1.4.1 Certification authority application

The IA may apply to the VICAL Provider for inclusion of its CA into the VICAL.

The respective IA point of contact shall provide all requested information by the VICAL Provider, according to the established processes and through the trusted communication channel. The provided information shall be authentic, correct, complete and truthful.

The Terms and Conditions of the VICAL Service, if any, shall be made available to the applicant by the VICAL Provider.

NOTE An IACA link certificate is an example of a trusted communication channel.

C.1.4.2 Certification authority application processing

The VICAL Provider shall validate the information submitted by the IA's point of contact. Should any of the acceptance checks and conditions fail, the VICAL Provider is reserved the right to terminate the application process.

The VICAL Provider shall keep records (see C.1.5.5) of the application analysis, processing, internal and external checks and results.

C.1.4.3 Certification authority application acceptance

If the application processing by the VICAL Provider approves the CA, it can be included into the VICAL for the corresponding issuing authority and published according to the timeline set by the VICAL Provider and agreed by the applicant.

C.1.4.4 Certification authority renewal

A CA renewal is understood as a request to include a new CA certificate with the same name, public key, and other information as the old one, but with a new, extended validity period and a new serial number.

CA renewals shall not be used.

C.1.4.5 Certification authority application re-key

A CA re-key is understood as a request to include a new CA certificate with a different public key (and serial number) while retaining the remaining contents of the old CA certificate. The new certificate may be assigned a different validity period, key identifiers, specify a different CRL distribution point, and/or be signed with a different key.

The old certificate may or may not be revoked, but shall not be further re-keyed, renewed, or modified.

VICAL Providers shall process CA re-keys as new CA applications and thus follow the definitions laid down in C.1.4.1 to C.1.4.3.

C.1.4.6 Certification authority application modification

Modifying a CA certificate is understood as a request to include a new certificate that has the same or a different key and a different serial number, and that differs in one or more other fields from the old certificate.

The old certificate may or may not be revoked, but shall not be further re-keyed, renewed, or modified.

VICAL Providers shall process CA modifications as new CA applications and thus follow the definitions laid down in C.1.4.1 to C.1.4.3.

C.1.4.7 Certification authority application suspension

The VICAL Provider shall suspend CA certificates in a timely manner based on authorized and validated requests from the issuing authority point of contact.

The VICAL Provider shall keep records (see C.1.5.5) of the analysis, processing, internal and external checks and results.

C.1.4.8 End of subscription

This policy does not define nor limits any particular form or nature of relation between VICAL Providers and issuing authorities, and VICAL Providers and subscribers. However, it assumes the existence of these Participants (see C.1.1.3) and some form of relation is established between them.

In the event of termination of the relation between the VICAL Provider and the issuing authority, the VICAL Provider is allowed to remove/suspend or maintain unchanged the corresponding IACA on the VICAL, according to the terms and conditions agreed between the parties, or at its sole discretion in its absence.

On the other side, in the event of termination of the relation between the VICAL Provider and a subscriber, the contents of the VICAL shall not be affected.

C.1.5 Facility, management and operational controls

C.1.5.1 Physical security controls

VICAL Provider's equipment shall be protected from unauthorized access while the cryptographic module (see C.1.6.2) is installed and activated. The VICAL Provider shall implement physical access controls to reduce the risk of equipment tampering even when the cryptographic module is not installed and activated. VICAL Provider cryptographic tokens shall be protected against theft, loss, and unauthorized use.

The following controls shall be fulfilled.

a) Physical access to components of the VICAL Provider's system whose security is critical to the provision of its VICAL services shall be limited to authorized individuals.

b) Controls shall be implemented to avoid loss, damage or compromise of assets and interruption to business activities.

c) Controls shall be implemented to avoid compromise or theft of information and information processing facilities.

d) Components that are critical for the secure operation of the VICAL service shall be located in a protected security perimeter with physical protection against intrusion, controls on access through the security perimeter and alarms to detect intrusion.

e) The facilities concerned with VICAL generation and management (i.e. CAs status lifecycle management) shall be operated in an environment which physically protects the services from compromise through unauthorized access to systems or data.

f) Every entry to the physically secure area shall be subject to independent oversight and non-authorized person shall be accompanied by an authorized person whilst in the secure area.

g) Every entry and exit shall be logged and such access log shall be inspected periodically.

h) Physical protection shall be achieved through the creation of clearly defined security perimeters (i.e. physical barriers) around the VICAL generation and management services.

i) Any parts of the premises shared with other organizations shall be outside the perimeter of the VICAL generation management services.

j) Physical and environmental security controls shall be implemented to protect the facility housing system resources, the system resources themselves, and the facilities used to support their operation.

k) The VICAL Provider's physical and environmental security policy for systems concerned with VICAL generation and management services shall address the physical access control, natural disaster protection, fire safety factors, failure of supporting utilities (e.g. power, telecommunications), structure collapse, plumbing leaks, protection against theft, breaking and entering, and disaster recovery.

l) Controls shall be implemented to protect against equipment, information, media and software relating to the VICAL Provider's services being taken off-site without authorization.

m) Other functions relating to VICAL Provider's operations may be supported within the same secured area provided that the access is limited to authorized personnel.

C.1.5.2 Procedural controls

VICAL Providers shall implement security measures in order to protect the authenticity, integrity and confidentiality of their data and the accurate functionality of their IT systems.

The following controls shall be fulfilled.

a) The VICAL Provider shall administer user access of operators, administrators and system auditors.

b) The administration shall include user account management and timely modification or removal of access.

c) Access to information and application system functions shall be restricted in accordance with the access control policy.

d) The VICAL Provider's system shall provide sufficient computer security controls for the separation of trusted roles identified in VICAL Provider's practices, including the separation of security administration and operation functions. Particularly, use of system utility programs shall be restricted and controlled.

e) VICAL Provider's personnel shall be identified and authenticated before using critical applications related to the service.

f) VICAL Provider's personnel shall be accountable for their activities.

g) Activation of the VICAL signing key shall be under at least dual control by authorized, trusted personnel such that one person alone cannot activate the VICAL creation system on his/her own.

C.1.5.3 Personnel controls

The VICAL Provider shall ensure that employees and contractors support the trustworthiness of the VICAL Provider's operations.

The following controls shall be fulfilled.

a) The VICAL Provider shall employ staff and, if applicable, subcontractors, who possess the necessary expertise, reliability, experience, and qualifications and who have received training regarding security and personal data protection rules as appropriate for the offered services and the job function.

b) VICAL Provider's personnel should be able to fulfil the requirement of "expert knowledge, experience and qualifications" through formal training and credentials, or actual experience, or a combination of the two.

c) This should include regular (at least every 12 months) updates on new threats and current security practices.

d) Appropriate disciplinary sanctions shall be applied to personnel violating VICAL Provider's policies or procedures.

e) Security roles and responsibilities, as specified in the VICAL Provider's information security policy, shall be documented in job descriptions or in documents available to all concerned personnel.

f) Trusted roles, on which the security of the VICAL Provider's operation is dependent, shall be clearly identified.

g) Trusted roles shall be named by the management.

h) Trusted roles shall be accepted by the management and the person to fulfil the role.

i) VICAL Provider's personnel (both temporary and permanent) shall have job descriptions defined from the view point of roles fulfilled with segregation of duties and least privilege, determining position sensitivity based on the duties and access levels, background screening and employee training and awareness.

j) Where appropriate, job descriptions shall differentiate between general functions and VICAL Provider's specific functions. These should include skills and experience requirements.

k) Personnel shall exercise administrative and management procedures and processes that are in line with the VICAL Provider's information security management procedures.

l) Managerial personnel shall possess experience or training with respect to the VICAL service that is provided, familiarity with security procedures for personnel with security responsibilities and experience with information security and risk assessment sufficient to carry out management functions.

m) All VICAL Provider's personnel in trusted roles shall be free from conflict of interest that might prejudice the impartiality of the VICAL Provider's operations.

n) Trusted roles shall include roles that involve the following responsibilities:

   1. Security Officers: overall responsibility for administering the implementation of the security practices;

   2. System Administrators: authorized to install, configure, maintain and recover the VICAL Provider's trustworthy systems for service management;

   3. System Operators: responsible for operating the VICAL Provider's trustworthy systems on a day-to-day basis. Authorized to perform system backup;

4. System Auditors: authorized to view archives and audit logs of the VICAL Provider's trustworthy systems.

o) VICAL Provider's personnel shall be formally appointed to trusted roles by senior management responsible for security requiring the principle of "least privilege" when accessing or when configuring access privileges.

p) Personnel shall not have access to the trusted functions until the necessary checks are completed.

C.1.5.4 Audit logging procedures

The VICAL Provider shall record and keep accessible for an appropriate period of time, including after the activities of the VICAL Provider have ceased, all relevant information concerning data issued and received by the VICAL Provider, in particular, for the purpose of providing evidence in legal proceedings and for the purpose of ensuring continuity of the service.

The following controls shall be fulfilled.

a) The confidentiality and integrity of current and archived records concerning operation of services shall be maintained.

b) Records concerning the operation of services shall be completely and confidentially archived in accordance with disclosed business practices.

c) Records concerning the operation of services shall be made available if required for the purposes of providing evidence of the correct operation of the services for the purpose of legal proceedings.

d) The precise time of significant VICAL Provider's environmental, key management and clock synchronization events shall be recorded.

e) The time used to record events as required in the audit log shall be synchronized with UTC at least once a day.

f) Records concerning services shall be held for a period of time as appropriate for providing necessary legal evidence and as notified in the VICAL Provider's terms and conditions.

g) The events shall be logged in a way that they cannot be easily deleted or destroyed (except if reliably transferred to long-term media) within the period of time that they are required to be held. Where possible, the security audit logs shall be automatically collected. Where this is not possible, a logbook, paper form, or other physical mechanism shall be used. All security audit logs, both electronic and non-electronic, shall be retained and made available during compliance audits.

C.1.5.4.1 Types of events recorded

The VICAL Provider shall record details of the actions taken to process a request and to issue a VICAL, including all information generated and documentation received in connection with the request; the time and date; and the personnel involved. The VICAL Provider shall make these records available to Auditors as proof of the VICAL Provider’s compliance with these requirements.

The VICAL Provider shall record at least the following events.

a) VICAL signing key lifecycle management events, including:

   1. key generation, backup, storage, recovery, archival, and destruction; and

   2. cryptographic device lifecycle management events.

b) VICAL and IA lifecycle management events, including:

   1. CA application, re-key requests, and suspension;

   2. all verification activities stipulated in these requirements;

3. date, time, phone number used, persons spoken to, and end results of verification telephone calls;
4. acceptance and rejection of CA applications; and
5. issuance of VICALs.

c) security events, including:

1. successful and unsuccessful VICAL Provider's system access attempts;
2. VICAL Provider's and security system actions performed;
3. security profile changes;
4. system crashes, hardware failures, and other anomalies;
5. firewall and router activities; and
6. entries to and exits from the VICAL Provider facility.

Log entries shall include the following elements:

d) date and time of entry;

e) identity of the person making the journal entry; and

f) description of the entry.

C.1.5.5 Records archival

The VICAL Provider shall retain the following for at least seven years after any CA (accepted or not) based on these records ceases to be valid:

a) log of all events relating to the life cycle of keys managed by the VICAL Provider's system;

b) documentation and other evidence as referred in C.1.4.2.

C.1.5.6 Key changeover

To minimize risk from compromise of a VICAL Provider's private signing key, that key may be changed often. From that time on, only the new key should be used to sign VICALs. If the old private key is necessary during a limited period of time to keep signing VICALs and allow the migration for Relying parties with legacy systems, the old key shall be retained and protected. Once the old private signing key is not needed anymore, it may be destroyed.

The VICAL Provider's signing key shall have a validity period as described in C.1.7.2.

When a VICAL Provider updates its private signature key and thus generates a new public key, the VICAL Provider shall notify all subscribers that rely on its respective VICALs that it has been changed. The VICAL Provider shall provide the new public key through secure means (e.g. trusted communication channel established with subscribers/relying parties, provision of key rollover certificates – new public key is signed by the old private key, and vice versa).

C.1.5.7 Compromise and disaster recovery

C.1.5.7.1 Incident and compromise handling procedures

System activities concerning access to IT systems, use of IT systems, and service requests shall be monitored.

The following controls shall be fulfilled.

a) Monitoring activities should take account of the sensitivity of any information collected or analysed.

b) Abnormal system activities that indicate a potential security violation, including intrusion into the VICAL Provider's network, shall be detected and reported as alarms.

c) The VICAL Provider shall monitor the following events:

   1. start-up and shutdown of the logging functions; and

   2. availability and utilization of needed services with the VICAL Provider's network.

d) The VICAL Provider shall act in a timely and co-ordinated manner in order to respond quickly to incidents and to limit the impact of breaches of security.

e) The VICAL Provider shall appoint trusted role personnel to follow up on alerts of potentially critical security events and ensure that relevant incidents are reported in line with the VICAL Provider's procedures.

f) The VICAL Provider shall establish procedures to notify the appropriate parties in line with the applicable regulatory rules of any breach of security or loss of integrity that has a significant impact on the VICAL service provided and on the personal data maintained therein within 24 hours of the breach being identified.

g) Where the breach of security or loss of integrity is likely to adversely affect a natural or legal person to whom the VICAL service has been provided, the VICAL Provider shall also notify the natural or legal person of the breach of security or loss of integrity without undue delay.

h) The VICAL Provider's systems shall be monitored including the monitoring or regular review of audit logs to identify evidence of malicious activity implementing automatic mechanisms to process the audit logs and alert personnel of possible critical security events.

i) The VICAL Provider shall address any critical vulnerability not previously addressed by the VICAL Provider, within a period of 48 hours after its discovery.

j) For any vulnerability, given the potential impact, the VICAL Provider may either choose to:

   1. create and implement a plan to mitigate the vulnerability; or

   2. document the factual basis for the VICAL Provider's determination that the vulnerability does not require remediation.

k) Incident reporting and response procedures shall be employed in such a way that damage from security incidents and malfunctions are minimized.

C.1.5.7.2 Computing resources, software, and/or data are corrupted

The following controls shall be fulfilled.

a) VICAL Provider's systems data necessary to resume CA operations shall be backed up and stored in safe places, preferably also remote, suitable to allow the VICAL Provider to timely go back to operations in case of incident/disasters.

b) Back-up copies of essential information and software should be taken regularly.

c) Adequate back-up facilities should be provided to ensure that all essential information and software can be recovered following a disaster or media failure.

d) Back-up arrangements should be regularly tested to ensure that they meet the requirements of business continuity plans.

e) Backup and restore functions shall be performed by the relevant trusted roles specified in C.1.5.3.

f) For information requiring dual control for management, for example keys, dual control shall be applied to recovery.

C.1.5.7.3 VICAL Provider private key compromise procedures

The following controls shall be fulfilled in case of a private key compromise:

a) The VICAL Provider's business continuity plan (or disaster recovery plan) shall address the compromise, loss or suspected compromise of a VICAL Provider's private key as a disaster.

b) The processes planned as per the previous requirement shall be in place.

c) Following a disaster, the VICAL Provider shall, where practical, take steps to avoid repetition of a disaster.

d) In the case of compromise as a minimum:

   1. the VICAL Provider shall inform the following of the compromise: all issuing authorities point of contacts and subscribers and other entities with which the VICAL Provider has agreements or other form of direct established relations, among which relying parties and VICAL Providers; and

   2. the VICAL Provider shall indicate that VICALs issued using this private key may no longer be valid.

Furthermore, the following controls shall be fulfilled in case of an algorithm compromise.

a) Should any of the algorithms, or associated parameters, used by the VICAL Provider or its issuing authorities point of contacts and/or subscribers become insufficient for its remaining intended usage, then the VICAL Provider shall inform all of them with whom it has agreement or other form of established relations.

b) Should any of the algorithms, or associated parameters, used by the VICAL Provider or its subscribers become insufficient for its remaining intended usage, then the VICAL Provider shall plan the transition to a new stronger algorithm and execute it the earliest possible time.

C.1.5.7.4 Business continuity capabilities after a disaster

The following controls shall be fulfilled.

a) The VICAL Provider shall define and maintain a continuity plan to enact in case of a disaster.

b) In the event of a disaster, including compromise of a private signing key or compromise of some other credential of the VICAL Provider, operations shall be restored within the delay established in the continuity plan, having addressed any cause for the disaster which may recur (e.g. a security vulnerability) with appropriate remediation measures.

C.1.5.8 VICAL termination

Potential disruptions to subscribers and relying parties shall be minimized as a result of the cessation of the VICAL Provider's services and, in particular, continued maintenance of information required to verify the correctness of VICAL services shall be provided.

Furthermore, the following controls shall be fulfilled.

a) The VICAL Provider shall have an up-to-date termination plan.

b) Before the VICAL Provider terminates its services, at least the following procedures apply.

   1. Before the VICAL Provider terminates its services, the VICAL Provider shall inform the following of the termination: all issuing authorities point of contacts and subscribers and other

entities with which the VICAL Provider has agreements or other form of established relations, among which relying parties and VICAL Providers.

2. Before the VICAL Provider terminates its services, the VICAL Provider shall terminate authorization of all subcontractors, if any, to act on behalf of the VICAL Provider in carrying out any functions relating to the processing and/or dissemination of the VICAL.

3. Before the VICAL Provider terminates its services, the VICAL Provider shall transfer obligations to a reliable party for maintaining all information necessary to provide evidence of the operation of the VICAL Provider for a reasonable period, unless it can be demonstrated that the VICAL Provider does not hold any such information. The minimum information set is composed of:

   i. registration information;

   ii. event log archives.

4. Before the VICAL Provider terminates its services, the VICAL Provider's private keys, including backup copies, shall be destroyed, or withdrawn from use, in a manner such that the private keys cannot be retrieved.

5. Before the VICAL Provider terminates its services, where possible VICAL Provider should make arrangements to transfer provision of VICAL services for its existing subscribers and Issuing Authorities point of contacts to another VICAL Provider.

c) The VICAL Provider shall have an arrangement to cover the costs to fulfil these minimum requirements in case the VICAL Provider becomes bankrupt or for other reasons is unable to cover the costs by itself, as far as possible within the constraints of applicable legislation regarding bankruptcy.

d) The VICAL Provider shall state in its practices the provisions made for termination of service. This shall include:

   1. notification of affected entities; and

   2. where applicable, transferring the VICAL Provider's obligations to other parties.

e) The VICAL Provider shall maintain or transfer to a reliable party its obligations to make available its public key or its history of VICALs to subscribers and relying parties for a reasonable period.

C.1.6 Technical security controls

C.1.6.1 Key pair generation and installation

Appropriate security controls shall be in place for the management of any cryptographic keys and any cryptographic devices throughout their lifecycle.

The following controls shall be fulfilled.

a) The VICAL signing key pair generation shall be undertaken in a physically secured environment (see C.1.5.1) by personnel in trusted roles (see C.1.5.3).

b) The VICAL Provider key pair used for signing VICALs shall be created under, at least, dual control.

c) The number of personnel authorized to carry out VICAL signing key pair generation shall be kept to a minimum and be consistent with the VICAL Provider's practices.

d) VICAL Signing key pair generation shall be performed using an algorithm as specified in C.1.7.2.

e) The selected key length and algorithm for VICAL signing key are specified in C.1.7.2.

f) Before expiration of its VICAL signer certificate which is used for signing VICALs, in case of continuing with the service, the VICAL Provider shall generate a new key pair and obtain a corresponding VICAL signer certificate, and shall apply all necessary actions to avoid disruption to the operations of any entity that may rely on the VICAL signer certificate.

g) Before expiration of its VICAL signer certificate, in case of continuing with the service, the new VICAL signer certificate shall also be issued and distributed in accordance with this document.

h) The operations described in f) and g) above should be performed with a suitable interval between certificate expiry date and the last VICAL signed to allow all parties that have relationships with the VICAL Provider (subscribers, relying parties, Issuing Authorities, etc.) to be aware of this key changeover and to implement the required operations to avoid inconveniences and malfunctions. This does not apply to a VICAL Provider which will cease its operations before its own VICAL signer certificate expiration date."

i) The VICAL Provider shall have a documented procedure for conducting generation of VICAL signing key pairs. Such procedure shall indicate, at least, the following:

   1. roles participating in the ceremony (internal and external from the organization);
   2. functions to be performed by every role and in which phases;
   3. responsibilities during and after the ceremony; and
   4. requirements of evidence to be collected of the ceremony.

j) The VICAL Provider shall produce a report proving that the ceremony, as in i) above, was carried out in accordance with the stated procedure and that the integrity and confidentiality of the key pair was ensured.

k) CA signature verification (public) keys of the VICAL signing certificate shall be available to subscribers and relying parties in a manner that assures the integrity of the CA public key and authenticates its origin.

C.1.6.2 Private key protection and cryptographic module engineering controls

The following controls shall be fulfilled.

a) VICAL Provider's signing key pair generation shall be carried out within a secure cryptographic device which is a trustworthy system which:

   1. is assured to EAL 4 or higher in accordance with ISO/IEC 15408 or equivalent national or internationally recognized evaluation criteria for IT security provided this is a security target or protection profile which meets the requirements of this document, based on a risk analysis and taking into account physical and other non-technical security measures; or
   2. meets the requirements identified in ISO/IEC 19790 or FIPS 140-2 level 3.

b) The secure cryptographic device shall be operated in its configuration as described in the appropriate certification guidance documentation or in an equivalent configuration which achieves the same security objective.

c) The VICAL private signing key shall be held and used within a secure cryptographic device meeting the requirements item a) and b) above.

d) If and when outside the secure cryptographic device, the VICAL signing private key shall be protected in a way that ensures the same level of protection as provided by the secure cryptographic device.

e) The VICAL signing private key may be backed up, stored and recovered only by personnel in trusted roles (see C.1.5.3) using, at least, dual control in a physically secured environment (see C.1.5.1).

f) The number of personnel authorized to carry out the VICAL signing private key back up, storage and recovery shall be kept to a minimum and be consistent with the VICAL Provider's practices.

g) Copies of the VICAL private signing keys shall be subject to the same or greater level of security controls as keys currently in use.

h) Where the VICAL signing private keys and any copies are stored in a dedicated secure cryptographic device, access controls shall be in place to ensure that the keys are not accessible outside this device.

i) The secure cryptographic device shall not be tampered with during shipment.

j) The secure cryptographic device shall not be tampered with while stored.

k) The secure cryptographic device shall be functioning correctly.

l) The CA private signing keys stored on the CA's secure cryptographic device shall be destroyed upon device retirement.

C.1.6.3 Other aspects of key pair management

The VICAL Provider shall use appropriately the VICAL private signing keys.

The following controls shall be fulfilled.

a) The VICAL Provider shall not use the VICAL signing private keys beyond the end of their life cycle.

b) VICAL signing key(s) used for generating VICALs as defined in C.1.7.1 shall not be used for any other purpose.

c) The VICAL signing keys shall only be used within physically secure premises.

d) The use of the VICAL's private key shall be compatible with the hash algorithm, the signature algorithm and signature key length used for generating VICALs (defined in C.1.7.2).

e) All copies, if any, of the VICAL signing private keys shall be destroyed at the end of their life cycle.

C.1.6.4 Activation data

The installation, activation and recovery of the VICAL signing key pairs in a secure cryptographic device shall require simultaneous control of at least two trusted employees, for example, using m-of-n authentication mechanisms.

Data used to unlock private keys shall be protected from disclosure by a combination of cryptographic and physical access control mechanisms. Activation data shall be either:

— memorized; or

— biometric in nature; or

— recorded and secured at the level of assurance associated with the activation of the cryptographic module, and shall not be stored with the cryptographic module.

C.1.6.5 Computer security controls

The VICAL Provider's system access shall be limited to authorized individuals.

The following controls shall be fulfilled.

a) Controls (e.g. firewalls) shall protect the VICAL Provider's internal network domains from unauthorized access including access by subscribers and third parties.

b) Firewalls should also be configured to prevent all protocols and accesses not required for the operation of the VICAL Provider.

c) Sensitive data shall be protected against being revealed through re-used storage objects (e.g. deleted files) being accessible to unauthorized users.

d) Local network components (e.g. routers) shall be kept in a physically and logically secure environment.

e) Local network components (e.g. routers) configurations shall be periodically checked for compliance with the requirements specified by the VICAL Provider.

f) The VICAL Provider shall enforce multi-factor authentication for all accounts capable of directly causing VICAL issuance.

g) Dissemination application shall enforce access control on attempts to add or delete VICALs and modify other associated information.

h) Continuous monitoring and alarm facilities shall be provided to enable the VICAL Provider to detect, register and react in a timely manner upon any unauthorized and/or irregular attempts to access its resources.

C.1.6.6 Life cycle security controls

The VICAL Provider shall use trustworthy systems and products that are protected against modification and ensure the technical security and reliability of the processes supported by them.

The following controls shall be fulfilled.

a) An analysis of security requirements shall be carried out at the design and requirements specification stage of any systems development project undertaken by the VICAL Provider or on behalf of the VICAL Provider to ensure that security is built into IT systems.

b) Change control procedures shall be applied for releases, modifications and emergency software fixes of any operational software and changes to the configuration which applies the VICAL Provider's security policy.

c) The procedures shall include documentation of the changes.

d) The integrity of VICAL Provider's systems and information shall be protected against viruses, malicious and unauthorized software.

e) Media used within the VICAL Provider's systems shall be securely handled to protect media from damage, theft, unauthorized access and obsolescence.

f) Media management procedures shall protect against obsolescence and deterioration of media within the period of time that records are required to be retained.

g) Procedures shall be established and implemented for all trusted and administrative roles that impact on the provision of services.

h) The VICAL Provider shall specify and apply procedures for ensuring that:

   1. security patches are applied within a reasonable time after they come available;

   2. security patches are not applied if they introduce additional vulnerabilities or instabilities that outweigh the benefits of applying them; and

   3. the reasons for not applying any security patches are documented.

i) Capacity demands shall be monitored and projections of future capacity requirements shall be made to ensure that adequate processing power and storage are available.

C.1.6.7 Network security controls

The VICAL Provider shall protect its network and systems from attack.

The following controls shall be fulfilled.

a) The VICAL Provider shall segment its systems into networks or zones based on risk assessment considering functional, logical, and physical (including location) relationship between trustworthy systems and services.

b) The VICAL Provider shall apply the same security controls to all systems co-located in the same zone.

c) The VICAL Provider shall restrict access and communications between zones to those necessary for the operation of the VICAL Provider.

d) The VICAL Provider shall explicitly forbid or deactivate not needed connections and services.

e) The VICAL Provider shall review the established rule set on a regular basis.

f) The VICAL Provider shall keep all systems that are critical to the VICAL Provider's operation in one or more secured zone(s).

g) The VICAL Provider shall separate dedicated network for administration of IT systems and VICAL Provider 's operational network.

h) The VICAL Provider shall not use systems used for administration of the security policy implementation for other purposes.

i) The VICAL Provider shall separate the production systems for the VICAL Provider's services from systems used in development and testing (e.g. development, test and staging systems).

j) The VICAL Provider shall establish communication between distinct trustworthy systems only through trusted channels that are logically distinct from other communication channels and provide assured identification of its end points and protection of the channel data from modification or disclosure.

k) If a high level of availability of external access to the VICAL service is required, the external network connection should be redundant to ensure availability of the services in case of a single failure.

l) The VICAL Provider shall undergo or perform a regular vulnerability scan on public and private IP addresses identified by the VICAL Provider and record evidence that each vulnerability scan was performed by a person or entity with the skills, tools, proficiency, code of ethics, and independence necessary to provide a reliable report.

m) The VICAL Provider shall undergo a penetration test on the VICAL Provider's systems at set up and after infrastructure or application upgrades or modifications that the VICAL Provider determines are significant.

n) The VICAL Provider shall record evidence that each penetration test was performed by a person or entity with the skills, tools, proficiency, code of ethics, and independence necessary to provide a reliable report.

o) The VICAL Provider shall maintain and protect all VICAL systems in at least a secure zone and shall implement and configure a security procedure that protects systems and communications between systems inside secure zones and high security zones.

p) The VICAL Provider shall configure all VICAL systems by removing or disabling all accounts, applications, services, protocols, and ports that are not used in the VICAL's operations.

q) The VICAL Provider shall grant access to secure zones and high security zones to only trusted roles.

r) The VICAL issuing system shall be in a high security zone.

C.1.6.8 Timestamping

The following controls shall be fulfilled.

a) Asserted times shall be accurate to within three minutes.

b) Electronic or manual procedures may be used to maintain system time.

c) Clock adjustments are auditable events.

C.1.7 VICAL and VICAL signer certificate profiles

C.1.7.1 VICAL CDDL profile

The VICAL profile shall use the following CDDL structure:

```
VICAL = {
    "version" : tstr                ; VICAL structure version, currently "1.0"
    "vicalProvider" : tstr          ; Identifies the VICAL provider
    "date" : tdate                  ; date-time of VICAL issuance
    ? "vicalIssueID" : uint         ; identifies the specific issue of the VICAL, shall be
unique and monotonically increasing
    ? "nextUpdate" : tdate          ; next VICAL is expected to be issued before this
date-time
    "certificateInfos" : [*CertificateInfo]
    ? "extensions" : Extensions     ; Can be used for proprietary extensions
    * tstr => any                   ; To be used for future extensions, all values are RFU
}

CertificateInfo = {
    "certificate" : bstr           ; DER-encoded X.509 certificate
    "serialNumber" : biguint        ; value of the  serial number field of the certificate
    "ski" : bstr                    ; value of the Subject Key Identifier field of the
certificate
    "docType" : [+ DocType]         ; DocType for which the certificate may be used as a
trust point
    ? "certificateProfile" : [+ CertificateProfile] ; Type of certificate
    ? "issuingAuthority" : tstr     ; Name of the certificate issuing authority
    ? "issuingCountry" : tstr       ; ISO3166-1 or ISO3166-2 depending on the issuing
authority
    ? "stateOrProvinceName" : tstr  ; State or province name of the certificate issuing
authority
    ? "issuer" : bstr               ; DER-encoded Issuer field of the certificate (i.e. the
complete Name structure)
    ? "subject" : bstr              ; DER-encoded Subject field of the certificate (i.e. the
complete Name structure)
    ? "notBefore" : tdate           ; value of the notBefore field of the certificate
    ? "notAfter" : tdate            ; value of the notAfter field of the certificate
    ? "extensions" : Extensions     ; Can be used for proprietary extensions
    * tstr => any                   ; To be used for future extensions, all values are RFU
}

Extensions = {* tstr => any}        ; Can be used for proprietary extensions

CertificateProfile = tstr          ; Uniform Resource Name (URN) according to RFC 8141
```

The CertificateInfo map contains the “docType” and “certificateProfile” key-value pairs. These pairs shall be used as follows.

— Each certificate in the VICAL is used by the associated issuing authority as a trust point to allow the verification of end-entity certificates in a given document ecosystem. The “docType” pair contains the document type(s) of the documents for which the current certificate is used. For the mDL ecosystem, the mDL doc type as specified in 7.1 shall be used. Relying parties should not attempt

to use the certificate as a trust point for an ecosystem whose document type is not listed in the "docType" key-value pair.

— The way in which a certificate may be used depends not only on the ecosystem for which it is a trustpoint, but optionally also on the certificate profile. The intended use(s) of each certificate is indicated by the "certificateProfile" pair. For an mDL IACA root certificate, the mDL IACA OID specified in B.1.1 shall be used.

EXAMPLE Within the mDL ecosystem the use of IACA root certificates is mandatory as the trust point for issuer data authentication. However, next to that optionally another type of (root) certificate may be used as the trust point for mdoc reader authentication. Within other document ecosystems, other security mechanisms may be used.

The `tdate` data types in the VICAL structure shall be encoded as specified in 7.1.

The `Extensions` structures may be used by the VICAL provider for prioprietary extensions.

The VICAL is encapsulated and signed by the untagged `COSE_Sign1` structure as defined in RFC 8152. Within the `COSE_Sign1` structure, the `payload` shall be `VICAL`. The `external_aad` field used in the `Sig_structure` shall be a bytestring of size zero.

The `alg` element (RFC 8152) shall be included as an element in the protected header. Other elements should not be present in the protected header.

The certificate containing the public key belonging to the private key used to sign the VICAL shall be included as an `x5chain` element as described in RFC: *CBOR Object Signing and Encryption (COSE): Headers for carrying and referencing X.509 certificates*. It shall be included as an unprotected header element.

The VICAL provider should use one of the signature algorithms for calculating the signature over the VICAL: "ES256", "ES384", "ES512" or "EdDSA". The VICAL provider should use one of the elliptic curves as specified in Table 22.

C.1.7.2 VICAL signer certificate profile

This certificate profile defines the VICAL signer certificates, establishing the minimum security parameters (key lengths, algorithms, policy IDs, etc.). CAs issuing certificates to VICAL Providers may choose to use equivalent or higher parameters, as well as other certificate fields and extensions that do not limit or reduce the overall security level. See Table C.1 for details.

This certificate uses an OID for the VICAL policy. The OID has the following definition:

id-mdl OBJECT IDENTIFIER ::= { iso(1) standard(0) 18013 5 }

id-mdl-kp OBJECT IDENTIFIER ::= { id-mdl 1 } - - arc for extended key purposes

id-mdl-kp-mdlVICAL OBJECT IDENTIFIER ::= { id-mdl-kp 8 } - - arc for mDL VICAL

Table C.1 — VICAL signer certificate profile

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Version	4.1.2.1	M		Shall be `v3`.
Serial number	4.1.2.2	M		Non-sequential positive, non-zero integer, shall contain at least 63 bits of output from a CSPRNG, should contain at least 71 bits of output from a CSPRNG, maximum 20 octets.
Signature	4.1.2.3	M		Value shall match the OID in the signature algorithm (below).
Issuer	4.1.2.4	M		According to the Certification Authority issuing the VICAL signer certificate
Validity	4.1.2.5	M		
Not before		M		Date on which the certificate validity period begins.
Not after		M		Maximum of 1 187 days after "Not before" date
Subject	4.1.2.6	M		Minimum required fields. Others may be present (e.g. Serial Number, State, Organization Unit, etc.).
Country (C)		M		Country code of jurisdiction of VICAL Provider. Encoded as `PrintableString`.
Organization (O)		M		Full registered name of the VICAL Provider. Encoded as `UTF8String`.
Common name (CN)		M		Name under which VICAL Provider operates VICAL service and is commonly known. Encoded as `UTF8String`.
Subject public key info	4.1.2.7	M		
algorithm		M		`1.2.840.10045.2.1` (Elliptic curve)
parameters		M		Implicitly specify curve parameters through an OID associated with one of the following curves specified in FIPS 186-4: `1.2.840.10045.3.1.7` (Curve P-256) `1.3.132.0.34` (Curve P-384) `1.3.132.0.35` (Curve P-521) Or one of the following curves specified in RFC 5639: `1.3.36.3.3.2.8.1.1.7` (brainpoolP256r1) `1.3.36.3.3.2.8.1.1.9` (brainpoolP320r1) `1.3.36.3.3.2.8.1.1.11` (brainpoolP384r1) `1.3.36.3.3.2.8.1.1.13` (brainpoolP512r1)
subjectPublicKey		M		Public key shall be encoded in uncompressed form.
X.509v3 extensions	4.2	M		Further extensions may be present if they are marked non-critical.

Key

Presence:

M mandatory

O optional

C conditional

Criticality:

C critical

NC not critical

Table C.1 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Authority key identifier	4.2.1.1	M	NC	
keyIdentifier		M		Same value as the subject key identifier of the issuer's certificate
Subject key identifier	4.2.1.2	M	NC	SHA-1 hash of the subject public key `BIT STRING` value (excluding tag, length, and number of unused bits).
Key usage	4.2.1.3	M	C	
Digital signature				0
Non-repudiation				1
Key encipherment				0
Data encipherment				0
Key agreement				0
Key certificate signature				0
CRL signature				0
Encipher only				0
Decipher only				0
Extended key usage	4.2.1.12	M	C	
Key usage		M		1.0.18013.5.1.8 (mdlVICAL)
CRLDistribution-Points	4.2.1.13	M	NC	The 'reasons' and 'cRL Issuer' fields shall not be used.
distributionPoint		M		URI for CRL distribution point
Private internet extensions				
Authority information access	4.2.2.1	C	NC	Conditional, shall be present if the IACA has an OCSP service.
Access description OCSP		M		
accessMethod		M		1.3.6.1.5.5.7.48.1 (OCSP)
accessLocation		M		URI for corresponding OCSP service

Key

Presence:

M mandatory

O optional

C conditional

Criticality:

C critical

NC not critical

Table C.1 *(continued)*

Certificate component	Section in RFC 5280	Presence	Criticality	Description
Signature algorithm	4.1.1.2	M		Options: `1.2.840.10045.4.3.2` (ECDSA-with SHA256) `1.2.840.10045.4.3.3` (ECDSA-with SHA384) `1.2.840.10045.4.3.4` (ECDSA with SHA512)
Signature value	4.1.1.3	M		Value according to the signature algorithm. By creating this signature, the CA certifies the binding between the public key material and the subject of the certificate, i.e. the VICAL Provider.
Key Presence: M mandatory O optional C conditional Criticality: C critical NC not critical				

C.1.8 Compliance audit and other assessment

It is beyond the scope of this Policy to establish an auditing scheme for VICAL Providers. Nevertheless, it is understood that VICAL Providers should be able to publicly demonstrate compliance to this Policy and the security requirements.

As a minimum, VICAL Providers shall conduct self-audits on a periodic basis (at least yearly) to assess compliance to this Policy.

An independent third-party assessment can be achieved by an VICAL Provider based on the following principles.

— Auditor qualification: VICAL Provider selects a competent certification body that meets the requirements of ISO/IEC 17021-1 or ISO/IEC 27006.

— Audit basis: the Audit is based on ISO/IEC 27001 and ISO/IEC 27002.

— Checking requirement realisation: the audit and control does not only check that procedural security controls are specified but also that they are adhered to in practice. This also includes the initial identity validation, the receipt of IACA applications and the suspension/removal procedure for IACAs.

— Iteration of audits and controls: audits and controls is performed at least every three years. The Auditing Body and the VICAL Provider carry out a review at least once a year by a team of one or more auditors to ensure on going compliance with this Policy.

— Being not conformant: in the event that an audit indicates that the VICAL Provider is not conformant to this Policy, or its certification becomes invalid or expires, the VICAL Provider notifies its point of contacts of Issuing Authorities and subscribers.

— Availability of audit results: the certificate of conformity can be made available to Issuing Authorities, subscribers, relying parties and other possible stakeholders.

A VICAL Provider should implement an Information Security Management System (ISMS) for its VICAL Service in accordance to ISO/IEC 27001. The ISMS is based on an ISMS policy of which its scope is defined by this Policy and, if applicable, the associated Practice Statement.

Annex D
(informative)

Data structure examples

D.1 General

This annex contains examples for different structures used in the document. Since CBOR results in binary structures, a diagnostic notation will be used together with the binary encoding, whenever CBOR examples are made in this annex. The diagnostic notation will indicate binary embedded cbor data as << >>.

D.2 Data elements

D.2.1 Driving privileges

CBOR data:

```
82a37576656869636c655f63617465676f72795f636f646561416a697373375655f64617465d903ec6a32303138
2d30382d30396b6578706972795f64617465d903ec6a323032342d31302d3230a37576656869636c655f636174
65676f72795f636f646561426a697373375655f64617465d903ec6a323031372d30322d32336b6578706972795f
64617465d903ec6a323032342d31302d3230
```

In diagnostic notation:

```
[
    {
        "vehicle_category_code": "A",
        "issue_date": 1004("2018-08-09"),
        "expiry_date": 1004("2024-10-20")
    },
    {
        "vehicle_category_code": "B",
        "issue_date": 1004("2017-02-23"),
        "expiry_date": 1004("2024-10-20")
    }
]
```

D.2.2 Age_over_nn

Table D.1 describes the different situations for the answers to an age_over_nn request given the presence of certain data elements on the mDL.

Table D.1 — Situations for answers to age_over_nn requests

		mDL holder actual age				
		19	**21**	**30**	**60**	**64**
	Attestation statements present on the mDL	age_over_21 = FALSE age_over_60 = FALSE	age_over_21 = TRUE age_over_60 = FALSE	age_over_21 = TRUE age_over_60 = FALSE	age_over_21 = TRUE age_over_60 = TRUE	age_over_21 = TRUE age_over_60 = TRUE
Business question	**Verifier request**	**Response (by mDL holder actual age)**				
Age equal to or above 18?	age_over_18	No response	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE
Age equal to or above 19?	age_over_19	No response	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE
Age equal to or above 20?	age_over_20	No response	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE
Age equal to or above 21?	age_over_21	age_over_21 = FALSE	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE
Age equal to or above 25?	age_over_25	age_over_21 = FALSE	No response	No response	age_over_60 is TRUE	age_over_60 is TRUE
Age equal to or above 30?	age_over_30	age_over_21 = FALSE	No response	No response	age_over_60 is TRUE	age_over_60 is TRUE
Age equal to or above 50?	age_over_50	age_over_21 = FALSE	No response	No response	age_over_60 is TRUE	age_over_60 is TRUE
Age equal to or above 60?	age_over_60	age_over_60 = FALSE	age_over_60 = FALSE	age_over_60 = FALSE	age_over_60 is TRUE	age_over_60 is TRUE
Age equal to or above 63?	age_over_63	age_over_60 = FALSE	age_over_60 = FALSE	age_over_60 = FALSE	No response	No response
Age equal to or above 64?	age_over_64	age_over_60 = FALSE	age_over_60 = FALSE	age_over_60 = FALSE	No response	No response
Age equal to or above 65?	age_over_65	age_over_60 = FALSE	age_over_60 = FALSE	age_over_60 = FALSE	No response	No response
Age below 18?	age_over_18[a]	No response	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE
Age below 19?	age_over_19[a]	No response	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE
Age below 20?	age_over_20[a]	No response	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE
Age below 21?	age_over_21[a]	age_over_21 = FALSE	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE	age_over_21 is TRUE
Age below 25?	age_over_25[a]	age_over_21 = FALSE	No response	No response	age_over_60 is TRUE	age_over_60 is TRUE
Age below 30?	age_over_30[a]	age_over_21 = FALSE	No response	No response	age_over_60 is TRUE	age_over_60 is TRUE
Age below 50?	age_over_50[a]	age_over_21 = FALSE	No response	No response	age_over_60 is TRUE	age_over_60 is TRUE
Age below 60?	age_over_60[a]	age_over_60 = FALSE	age_over_60 = FALSE	age_over_60 = FALSE	age_over_60 is TRUE	age_over_60 is TRUE

[a] The business question "Is the mDL holder under the age of x" is converted into the question "Is the mDL holder of age x or older", since this is the question the data structure is designed to answer. A response indicating that the mDL holder is of age x or older logically can be converted into the statement "the mDL holder is not under the age of x". Likewise, a response indicating that the mDL holder is not of age x or older can logically be converted into the statement "the mDL holder is under the age of x".

Table D.1 *(continued)*

		mDL holder actual age				
		19	**21**	**30**	**60**	**64**
	Attestation statements present on the mDL	age_over_21 = FALSE age_over_60 = FALSE	age_over_21 = TRUE age_over_60 = FALSE	age_over_21 = TRUE age_over_60 = FALSE	age_over_21 = TRUE age_over_60 = TRUE	age_over_21 = TRUE age_over_60 = TRUE

Business question	**Verifier request**	**Response (by mDL holder actual age)**				
Age below 63?	age_over_63 [a]	age_over_60 = FALSE	age_over_60 = FALSE	age_over_60 = FALSE	No response	No response
Age below 64?	age_over_64 [a]	age_over_60 = FALSE	age_over_60 = FALSE	age_over_60 = FALSE	No response	No response
Age below 65?	age_over_65 [a]	age_over_60 = FALSE	age_over_60 = FALSE	age_over_60 = FALSE	No response	No response

[a] The business question "Is the mDL holder under the age of x" is converted into the question "Is the mDL holder of age x or older", since this is the question the data structure is designed to answer. A response indicating that the mDL holder is of age x or older logically can be converted into the statement "the mDL holder is not under the age of x". Likewise, a response indicating that the mDL holder is not of age x or older can logically be converted into the statement "the mDL holder is under the age of x".

D.3 Device engagement

D.3.1 Device engagement structure

This is an example of a device engagement structure for QR device engagement.

CBOR data:

```
a30063312e30018201d818584ba4010220012158205a88d182bce5f42efa59943f33359d2e8a968ff289d93e5fa
444b624343167fe225820b16e8cf858ddc7690407ba61d4c338237a8cfcf3de6aa672fc60a557aa32fc6702818
30201a300f401f50b5045efef742b2c4837a9a3b0e1d05a6917
```

In diagnostic notation:

```
{
  0: "1.0",
  1:
  [
    1,
    24(<<
      {
        1: 2,
        -1: 1,
        -2: h'5A88D182BCE5F42EFA59943F33359D2E8A968FF289D93E5FA444B624343167FE',
        -3: h'B16E8CF858DDC7690407BA61D4C338237A8CFCF3DE6AA672FC60A557AA32FC67'
      }
    >>)
  ],
  2:
  [
    [
      2,
      1,
      {
        0: false,
        1: true,
        11: h'45EFEF742B2C4837A9A3B0E1D05A6917'
      }
    ]
  ]
}
```

D.3.2 NFC Handover request

A handover request example containing a request for NFC, BLE and Wi-Fi Aware transmission technologies:

```
9102254872159102026372010211020461630101300011020661630103 6e6663005102046163010157001a201e
016170706c69636174696f6e2f766e642e626c7565746f6f74682e6c652e6f6f6230081b28078080bf2801021c
021107c832fff6d26fa0beb34dfcd555d4823a1c11010369736f2e6f72673a31383031333a6e66636e6663015a1
72b016170706c69636174696f6e2f766e642e7766612e6e616e57030101032302001324fec9a70b97ac9684a4e
326176ef5b981c5e8533e5f00298cfccbc35e700a6b020414
```

D.3.3 NFC Handover select

A handover select example as a response to the D.3.2 example. The BLE transmission technology is selected:

```
91020f487315d1020961630101300104 6d646f631a200c016170706c69636174696f6e2f766e642e626c756574
6f6f74682e6c652e6f6f6230081b28128b37282801021c015c1e580469736f2e6f72673a31383031333a646576
696365656e676167656d656e746d646f63a20063312e30018201d818584ba4010220012158205a88d182bce5f4
2efa59943f33359d2e8a968ff289d93e5fa444b624343167fe225820b16e8cf858ddc7690407ba61d4c338237a8
cfcf3de6aa672fc60a557aa32fc67
```

D.4 Data retrieval

D.4.1 Device retrieval

D.4.1.1 mdoc request

CBOR data:

```
a26776657273696f6e63312e306b646f635265717565573747381a26c6974656d7352657175657374d8185893
a267646f6354797065756f72672e69736f2e31383031332e352e312e6d444c6a6e616d655370616365736a171
6f72672e69736f2e31383031332e352e31a66b66616d696c795f6e616d65f56f646f63756d656e745f6e756d
626572f57264726976696e675f70726976696c65676573f56a69737375655f64617465f56b6578706972795f
64617465f568706f727472616974f46a7265616465724175746884 43a10126a118215901b7308201b3308201
58a003020102021475527 15f6add323d4934a1ba175dc945755d8b50300a06082a8648ce3d0403023016311430
1206035504030c0b726561646572207 26f6f74301e170d323031303031303030303030305a170d3233313233313
030303030305a3011310f300d06035504030c0672656164657230593013 06072a8648ce3d020106082a8648ce
3d03010703420004f8912ee0f912b6be683ba2fa0121b2630e601b2b628dff3b44f6394eaa9abdbcc2149d29d6f
f1a3e091135177e5c3d9c57f3bf839761eed02c64dd82ae1d3bbfa38188308185301c0603551d1f04153013301
1a00fa00d820b6578616d706c652e636f6d301d0603551d0e04160414f2dfc4acafc5f30b464fada20bfcd533a
f5e07f5301f0603551d23041830168014cfb7a881baea5f32b6fb91cc29590c50dfac416e300e0603551d0f010
1ff040403020780301506 03551d250101ff040b3009060728818c5d050106300a06082a8648ce3d040302034900
3046022100fb9ea6b686fd7ea2f0234858ff8328b4efef6a1ef71ec4aae4e307206f9214930221009b94f0d739
dfa84cca29efed529dd4838acfd8b6bee212dc6320c46feb839a35f658401f3400069063c189138bdcd2f6314
27c589424117c9ec26cebcacacfcdb9695d28e99953becabc4e30ab4efacc839a81f9159933d192527ee91b44
9bb7f80bf
```

In diagnostic notation:

```
{
  "version": "1.0",
  "docRequests":
  [
    {
      "itemsRequest":
      24(<<
        {
          "docType": "org.iso.18013.5.1.mDL",
          "nameSpaces":
          {
            "org.iso.18013.5.1":
            {
              "family_name": true,
              "document_number": true,
              "driving_privileges": true,
              "issue_date": true,
```

```
              "expiry_date": true,
              "portrait": false
            }
          }
        }
      >>),
      "readerAuth":
      [
        << {1: -7} >>
        {
          33:
h'308201B330820158A00302010202147552715F6ADD323D4934A1BA175DC945755D8B50300A06082A8648CE3D
040302301631143012060355040304030C0B726561646572207261F6F74301E170D323031303031303031303030305A17
0D323331323331303030303030305A3011310F300D06035504030C0672656164657230593013060072A8648CE3D02
0106082A8648CE3D030107034200004F8912EE0F912B6BE683BA2FA0121B2630E601B2B628DFF3B44F639AEAA9A
BDBCC2149D29D6FF1A3E091135177E5C3D9C57F3BF839761EED02C64DD82AE1D3BBFA38188308185301C0603551
D1F041530133011A00FA00D820B6578616D706C652E636F6D301D0603551D0E04160414F2DFC4ACAFC5F30B46
4FADA20BFCD533AF5E07F5301F0603551D23041830168014CFB7A881BAEA5F32B6FB91CC29590C50DFAC416E30
0E0603551D0F0101FF040403020780301506035 51D250101FF040B3009060728818C5D05010680 0A06082A8648
CE3D040302034900304602210 0FB9EA3B686FD7EA2F0234858FF8328B4EFEF6A1EF71EC4AAE4E307206F921493
0221009B94F0D739DFA84CCA29EFED529DD4838ACFD8B6BEE212DC6320C46FEB839A35'
        },
        null,
h'1F3400069063C189138BDCD2F631427C589424113FC9EC26CEBCACACFCDB9695D28E99953BECABC4E30AB4EF
ACC839A81F9159933D192527EE91B449BB7F80BF'
      ]
    }
  ]
}
```

D.4.1.2 mdoc response

CBOR data:

```
a36776657273696f6e63312e3069646f63756d656e747381a367646f6354797065756f72672e69736f2e313
83031332e352e312e6d444c6c697373756572536976e6e4a26a6e616d65537061636573a1716f72672e69
736f2e31383031332e352e3186d8185863a468646967657374494400672616e646f6d58208798645b20ea
200e19ffabac92624bee6aec63aceedecfb1b80077d2bfc20e971656c656d656e744964656e74696669657
26b66616d696c795f6e616d656c656c656d656e7456616c756563446f65d818586ca468646967657374494
4036672616e646f6d5820b23f627e8999c706dfc0a4ed98ad74af988af619b4bb078b89058553f44615d7
1656c656d656e744964656e7469666965726a697373375655f646174656c656c656d656e7456616c7565d90
3ec6a323031392d31302d3230d818586da468646967657374494404667 2616e646f6d5820c7ffa307e5de92
1e67ba5878094787e8807ac8e7b5b3938d2ce80f00f3e9abaf71656c656d656e744964656e746966696572
6b6578706972795f646174656c656c656d656e7456616c7565d903ec6a323032342d31302d3230d818586d
a468646967657374494407667261646f6d582026052a42e5880557a806c1459af3fb7eb505d378156632
9d0b604b845b5f9e6871656c656d656e744964656e7469666965726f646f63756d656e745f6e756d626572
6c656c656d656e7456616c756669313233343536373839d818590471a468646967657374494408667 2616e
646f6d5820d094dad764a2eb99deb5210e9d899643efbd1d069cc311d3295516ca0b024412d71656c656d65
6e744964656e7469666965726870 6f7274726169746c656c656d656e7456616c7565590412ffd8ffe000104a
4649460001010100900090000ffdb004300130d0e110e0c13110f11151413171d301f1d1a1a1d3a2a2c233
0453d4947443d43414e66d5d4c51685241435f82606871757b7c7b4a5c869085778f6d787b76ffdb004301
1415151d191d381f1f38764f434f767676767676767676767676767676767676767676767676767676767676
767676767676767676767676767676767 6767676767676ffc000110800180064030122000211010311 01ffc4001
b000003010003010000000000000000000000000506040102030 7ffc400321000010303030205020309000000000
000102030400051106122113311415516171224107811a1635425273 91b2c1f1ffc400150101010000000000
000000000000000000001ffc4001a1101010100030100000000000000000000000014111213161ffda000c03010
002110311003f00a5bbde22da2329c7d692bc7d0d03f52cfb0ff75e7a7ef3e7709723a1d0dae146ddfbb3c039ce
07ad2bd47a7e32dbb8dd1d52d6ef4b284f64a480067dfb51f87ffb95ff00eb9ff14d215de66af089ce44b7dbde9cb
6890a2838eddf18078f7add62d411ef4db9b10a65d6b95a147381ea0d495b933275fe6bba75c114104a8ba4104
13e983dff004f5af5d34b4b4cde632d0bf1fd1592bdd91c6411f3934c2fa6af6b54975d106dcf4a65ae56e85600
1ebc03c7ce29dd9eef1ef10fc447dc9da76ad2aee93537a1ba7e4f70dd8eff0057c6dffb5e1a19854a83758e5452
8750946ec6704850cd037bceb08b6d7d2cc76d3317fc7b5cc04fb6707269c5c6e0c5b60ae549242123b0e493f6
02a075559e359970d98db89525456b51c951c8afa13ea8e98e3c596836783d5c63f5a61a99fdb7290875db4be8
8ab384bbbbbfc7183fdeaa633e8951db7da396dc48524fb1a8bd611a5aa2a2432f30ab420a7a6d3240c718cf03
1fa9ef4c9ad550205aa02951df4a1d6c8421b015b769db8c9229837ea2be8b1b0d39d0eba9c51484efdb8c0efd
8d258daf3c449699f2edbd4584e7af9c64e3f96b9beb28d4ac40931e6478c8e76a24a825449501d867d2b1dcde
bae99b9c752ae4ecd6dde4a179c1c1e460938f9149ef655e515c03919a289cb3dca278fb7bf177f4faa829dd8c
e3f2ac9a7ecde490971fafd7dce15eed9b71c018c64fa514514b24e8e4f8c5c9b75c1e82579dc1233dfec08238
f6add62d391acc1c5256a79e706d52d431c7a0145140b9fd149eb3a60dc5e88cbbc2da092411e9dc71f39a7766
b447b344e847dcac9dcb5abba8d145061d43a6fcf1e65cf15d0e90231d3dd9cfe62995c6dcc5ca12a2c904a15f
71dd27d451453e09d1a21450961cbb3ea8a956433b781f1ce33dfed54f0e2b50a2b71d84ed6db18028a28175f7
```

```
4fc6bda105c529a791c25c4f3c7a11f71586268f4a66b726e33de9ea6f1b52b181c760724e47b514520a5a28a2
83ffd9d81858ffa4686469676573744944096672616e646f6d58204599f81beaa2b20bd0ffcc9aa03a6f985befab3
f6beaffa41e6354cdb2ab2ce471656c656d656e744964656e7469666965727264726976696e675f70726976696c
656765736c656c656d656e7456616c756582a3757665686963 6c655f63617465676f72795f636f646561416a69
737375655f64617465d903ec6a323031382d30382d30396b6578706972795f64617465d903ec6a323032342d31
302d3230a37576656869636c655f63617465676f72795f636f646561426a69737375655f64617465d903ec6a32
3031372d30322d32336b6578706972795f64617465d903ec6a323032342d31302d32306a697373756572417574
688443a10126a118215901f3308201ef30820195a00302010202143c4416eed784f3b413e48f56f075abfa6d87
eb84300a06082a8648ce3d0403023023311430120603550403 0c0b75746f7069612069616361310b3009060355
040613025553301e170d323031303031303030303030305a170d323131303031303030303030305a30213112301006
035504030c0975746f706961206473310b300906035504061302555330595301306072a8648ce3d020106082a86
48ce3d03010703420004ace7ab7340e5d9648c5a72a9a6f56745c7aad436a03a43efea77b5fa7b88f0197d57d8
983e1b37d3a539f4d588365e38cbbf5b94d68c547b5bc8731dcd2f146ba381a83081a5301e0603551d12041730
1581136578616d706c65406578616d706c652e636f6d301c0603551d1f041530133011a00fa00d820b6578616d
706c652e636f6d301d0603551d0e0416041414e29017a6c35621ffc7a686b7b72db06cd12351301f0603551d230
4183016801454fa2383a04c28e0d930792261c80c4881d2c00b300e0603551d0f0101ff040403020780301506603
551d250101ff040b3009060728818c5d050102300a06082a8648ce3d040302034800304502210097717ab901674
0c8d7bcdaa494a62c053bbdecce1383c1aca72ad08dbc04cbb202203bad859c13a63c6d1a87d814d43e2425ca
f90d422422c04a8ee0304c0d3a68d5903a2d81859039da66776657273696f6e63312e306f646967657374416c6
76f726974686d675348412d3235366c76616c756544696765737473a2716f72672e69736f2e31383031332e352
e31ad00582075167333b47b6c2bfb86eccc1f438cf57af055371ac55e1e359e20f254adcebf01582067e539d61
39ebd131aef441b445645dd831b2b375b390ca5ef6279b205ed457102582033943 72ddb78053f36d5d869780e6
1eda313d44a392092ad8e0527a2fbfe55ae0358202e35ad3c4e514bb67b1a9db51ce74e4cb9b7146e41ac52dac
9ce86b8613db555045820ea5c3304bb7c4a8dcb51c4c13b65264f8455413413 42093cca786e058fac2d5905582
0fae487f68b7a0e87a749774e56e9e1dc3a8ec7b77e490d21f0e1d3475661aa1d0658207d83e507ae77db815de
4d803b88555d0511d894c897439f5774056416a1c7533075820f0549a145f1cf75cbeeffa881d4857dd438d627c
f32174b1731c4c38e12ca936085820b68c8afcb2aaf7c581411d2877de155be2eb121a42bc9ba5b7312377e06
8f660958200b3587d1dd0c2a07a35bfb120d99a0abfb5df56865bb7fa15cc8b56a66df6e0c0a5820c98a170cf3
6e11abb724e98a75a5343dfa2b6ed3df2ecfbb8ef2ee55dd41c8810b5820b57dd036782f7b14c6a30faaaae6cc
d5054ce88bdfa51a016ba75eda1edea9480c5820651f8736b18480fe252a03224ea087b5d10ca5485146c67c74
ac4ec3112d4c3a746f72672e69736f2e31383031332e352e312e5553a4005820d80b83d25173c484c5640610ff1
a31c949c1d934bf4cf7f18d5223b15dd4f21c0158204d80e1e2e4fb246d97895427ce7000bb59bb24c8cd003ec
f94bf35bbd2917e340258208b331f3b685bca372e85356a25c9484ab7afcdf0d2233105511f778d98c2f544035
820c343af1bd1690715439161aba73702c474abf992b20c9fb55c36a336ebe01a876d6465766963654b6579496
e666fa1696465766963654b6579a401022001215809 6313d6c63e24e3372742bfdb1a33ba2c897dcd68ab8c75
3e4fbd48dca6b7f9a2258201fb3269edd418857de1b39a4e4a44b92fa484caa722c228288f01d0c03a2c3d6676
46f6354797065756f72672e69736f2e31383031332e352e312e6d444c6c76616c6964697479496e666fa366736
9676e6564c074323032302d31302d30315431333a33303a30325a6976616c696446726f6dc074323032302d313
02d30315431333a33303a30325a6a76616c6964556e74696cc074323032312d31302d30315431333a33303a303
25a584059e64205df1e2f708dd6db0847aed79fc7c0201d80fa55badcaf2e1bcf5902e1e5a62e4832044b890ad
85aa53f129134775d733754d7cb7a4 3766aeff13cb2e6c6465766963655369676e6564a26a6e616d6553706163
6573d81841a06a6465766963654175 468a1696465766963654d61638443a10105a0f65820e99521a85ad7891b
806a07f8b5388a332d92c189a7bf293ee1f543405ae6824d6673746174757300
```

In diagnostic notation:

```
{
  "version": "1.0",
  "documents":
  [
    {
      "docType": "org.iso.18013.5.1.mDL",
      "issuerSigned":
      {
        "nameSpaces":
        {
          "org.iso.18013.5.1":
          [
            24(<<
              {
                "digestID": 0,
                "random": h'8798645B20EA200E19FFABAC92624BEE6AEC63ACEEDECFB1B80077D22BFC
20E9',
                "elementIdentifier": "family_name",
                "elementValue": "Doe"
              }
            >>),
            24(<<
              {
                "digestID": 3,
                "random": h'B23F627E8999C706DF0C0A4ED98AD74AF988AF619B4BB078B89058553F446
15D',
```

```
                "elementIdentifier": "issue_date",
                "elementValue": 1004("2019-10-20")
              }
            >>),
            24(<<
              {
                "digestID": 4,
                "random": h'C7FFA307E5DE921E67BA5878094787E8807AC8E7B5B3932D2CE80F00F3E9A
BAF',
                "elementIdentifier": "expiry_date",
                "elementValue": 1004("2024-10-20")
              }
            >>),
            24(<<
              {
                "digestID": 7,
                "random": h'26052A42E5880557A806C1459AF3FB7EB505D3781566329D0B604B945B5
F9E68',
                "elementIdentifier": "document_number",
                "elementValue": "123456789"
              }
            >>),
            24(<<
              {
                "digestID": 8,
                "random": h'D094DAD764A2EB9DEB5210E9D899643EFBD1D069CC311D3295516CA0B0244
12D',
                "elementIdentifier": "portrait",
                "elementValue": h'FFD8FFE000104A4649460001010100900090000FFDB004300130D0E
110E0C13110F11151413171D301F1D1A1A1D3A2A2C2330453D4947443D43414C566D5D4C51685241435F826068
71757B7C7B4A5C869085778F6D787B76FFDB0043011415151D191D381F1F38764F4
34F7676767676767676767676767676767676767676767676767676767676767676767676767
6767676767676FFC000110800180064030122000211010311101FFC4001B0000030100030100000000000000000
0000005060401020307FFC400321000010303030205020309000000000001020304000511061221133114155
1617122410781A1163542527391B2C1F1FFC400150101010100000000000000000000000000001FFC4001A110
1010100030100000000000000000000001411121316151FFDA000C03010002110311003F00A5BBDE22DA2329C7D
692BC7D0D03F52CFB0FF75E7A7EF3E7709723A1D0DAE146DDFBB3C039CE07AD2BD47A7E32DBB8DD1D52D6EF4B2
84F64A480067DFB51F87FFB95FF00EB9FF14D215DE66AF089CE44B7DBDE9CB6890A2838EDDF18078F7ADD62D41
1EF4DB9B10A65D6B95A147381EA0D495B933275FE6BBA75C114104A8BA410413E983DFF004F5AF5D34B4B4CDE6
32D0BF1FD1592BDD91C6411F3934C2FA6AF6B54975D106DCF4A65AE56E856001EBC03C7CE29DD9EEF1EF10FC44
7DC9DA76AD2AEE93537A1BA7E4F70DD8EFF005C6DFFB5E1A19854A83758E54528750946EC6704850CD037BCEB
08B6D7D2CC76D3317FC7B5CC04FB6707269C5C6E0C5B60AE549242123B0E493F602A075559E359970D98DB8952
5456B51C951C8AFA13EA8E98E3C5968367833D5C63F5A61A99FDB7290875DB4BE88AB384BBBBBFC7183FDEAA633
E8951DB7DA396DC48524FB1A8BD611AAA2A2432F30AB420A7A6D3240C718CF031FA9EF4C9AD550205AA02951D
F4A1D6C8421B015B769DB8C9229837EA2BE8B1B0D39D0EBA9C51484EFDB8C0EFD8D258DAF3C449699F2EDBD458
4E7AF9C64E3F96B9BEB28D4AC4093E6478C8E76A24A825449501D867D2B1DCDEBAE99B9C752AE4ECD6DDE4A17
9C1C1E460938F9149EF655E515C03919A289CB3DCA278FB7BF177F4FAA829DD8CE3F2AC9A7ECDE490971FAFD7D
CE15EED9B71C018C64FA514514B24E8E4F8C5C9B75C1E82579DC1233DFEC08238F6ADD62D391ACC1C5256A79E7
06D52D431C7A0145140B9ED149EB3A60DC5E88CBBC2DA092411E9DC71F39A7766B447B344E847DCAC9DCB5ABBA
8D145061D43A6FCF1E65CD15D0E90231D3DD9CFE62995C6DCC5CA12A2C904A15F71DD27D451453E09D1A214509
61CBB3EA8A956433B7B4F1CE33DFED54F0E2B50A2B71D84ED6DB18028A28175F74FC6BDA105C529A791C25C4F3
C7A11F71586268F4A66B726E33DE9EA6F1B52B181C760724E47B514520A5A28A283FFD9'
              }
            >>),
            24(<<
              {
                "digestID": 9,
                "random": h'4599F81BEAA2B20BD0FFCC9AA03A6F985BEFAB3F6BEAFFA41E6354CDB2AB
2CE4',
                "elementIdentifier": "driving_privileges",
                "elementValue":
                [
                  {
                    "vehicle_category_code": "A",
                    "issue_date": 1004("2018-08-09"),
                    "expiry_date": 1004("2024-10-20")
                  },
                  {
                    "vehicle_category_code": "B",
                    "issue_date": 1004("2017-02-23"),
                    "expiry_date": 1004("2024-10-20")
```

```
                }
              ]
            }
          >>)
        ]
      },
      "issuerAuth":
      [
        << {1: -7} >>,
        {
          33:
h'308201EF30820195A00302010202143C4416EED784F3B413E48F56F075ABFA6D87EB84300A06082A8648CE3D
040302302331143012060355040304030C0B75746F7069612069616361310B3009060355040613025553301E170D32
303130303130303030305A170D3231313030313030303030305A302131123010060355040304030C0975746F7069
61206473310B3009060355040613025553305930130607 2A8648CE3D020106082A8648CE3D030107034200 04AC
E7AB7340E5D9648C5A72A9A6F56745C7AAD436A03A43EFEA77B5FA7B88F0197D57D8983E1B37D3A539F4D58836
5E38CBBF5B94D68C547B5BC8731DCD2F146BA381A83081A5301E0603551D120417301581136578616D706C6540
6578616D706C652E636F6D301C0603551D1F041530133011A00FA00D820B6578616D706C652E636F6D301D0603
551D0E0416041414E29017A6C35621FFC7A686B7B72DB06CD12351301F0603551D2304183016801454FA2383A0
4C28E0D930792261C80C4881D2C00B300E0603551D0F0101FF040403020780301506035 51D250101FF040B3009
060728818C5D050102300A06082A8648CE3D040302034800304502210097717AB9016740C8D7BCDAA494A62C05
3BBDECCE1383C1ACA72AD08DBC04CBB202203BAD859C13A63C6D1AD67D814D43E2425CAF90D422422C04A8EE03
04C0D3A68D'
        },
        <<
          24(<<
            {
              "version": "1.0",
              "digestAlgorithm": "SHA-256",
              "valueDigests":
              {
                "org.iso.18013.5.1":
                {
                  0: h'75167333B47B6C2BFB86ECCC1F438CF57AF055371AC55E1E359E20F254ADC
EBF',
                  1: h'67E539D6139EBD131AEF441B445645DD831B2B375B390CA5EF6279B20
5ED4571',
                  2: h'3394372DDB78053F36D5D869780E61EDA313D44A392092AD8E0527A2FBFE5
5AE',
                  3: h'2E35AD3C4E514BB67B1A9DB51CE74E4CB9B7146E41AC52DAC9CE86B861
3DB555',
                  4: h'EA5C3304BB7C4A8DCB51C4C13B65264F845541341342093CCA786E058FA
C2D59',
                  5: h'FAE487F68B7A0E87A749774E56E9E1DC3A8EC7B77E490D21F0E1D3475661A
A1D',
                  6: h'7D83E507AE77DB815DE4D803B88555D0511D894C897439F5774056416
A1C7533',
                  7: h'F0549A145F1CF75CBEEFFA881D4857DD438D627CF32174B1731C4C38E1
2CA936',
                  8: h'B68C8AFCB2AAF7C581411D2877DEF155BE2EB121A42BC9BA5B7312377E0
68F66',
                  9: h'0B3587D1DD0C2A07A35BFB120D99A0ABFB5DF56865BB7FA15CC8B56A66DF6
E0C',
                  10: h'C98A170CF36E11ABB724E98A75A5343DFA2B6ED3DF2ECFBB8EF2EE55DD
41C881',
                  11: h'B57DD036782F7B14C6A30FAAAAE6CCD5054CE88BDFA51A016BA75EDA1E
DEA948',
                  12: h'651F8736B18480FE252A03224EA087B5D10CA5485146C67C74AC4EC3112D4
C3A'
                },
                "org.iso.18013.5.1.US":
                {
                  0: h'D80B83D25173C484C5640610FF1A31C949C1D934BF4CF7F18D5223B15DD4F
21C',
                  1: h'4D80E1E2E4FB246D97895427CE7000BB59BB24C8CD003ECF94BF35BBD29
17E34',
                  2: h'8B331F3B685BCA372E85351A25C9484AB7AFCDF0D2233105511F778D98
C2F544',
                  3: h'C343AF1BD1690715439161ABA73702C474ABF992B20C9FB55C36A336EBE01A87'
                }
              },
```

```
                "deviceKeyInfo":
                {
                  "deviceKey":
                  {
                    1: 2,
                    -1: 1,
                    -2: h'96313D6C63E24E3372742BFDB1A33BA2C897DCD68AB8C753E4FBD48DCA6B7
F9A',
                    -3: h'1FB3269EDD418857DE1B39A4E4A44B92FA484CAA722C228288F01D0C03A2
C3D6'
                  }
                },
                "docType": "org.iso.18013.5.1.mDL",
                "validityInfo":
                {
                  "signed": 0("2020-10-01T13:30:02Z"),
                  "validFrom": 0("2020-10-01T13:30:02Z"),
                  "validUntil": 0("2021-10-01T13:30:02Z")
                }
              }
            >>)
          >>,
          h'59E64205DF1E2F708DD6DB0847AED79FC7C0201D80FA55BADCAF2E1BCF5902E1E5A62E4832044B
890AD85AA53F129134775D733754D7CB7A413766AEFF13CB2E'
        ]
      },
      "deviceSigned":
      {
        "nameSpaces": 24(<< {} >>),
        "deviceAuth":
        {
          "deviceMac":
          [
            << {1: 5} >>,
            {},
            null,
            h' E99521A85AD7891B806A07F8B5388A332D92C189A7BF293EE1F543405AE6824D'
          ]
        }
      }
    }
  ],
  "status": 0
}
```

D.4.2 Server retrieval

D.4.2.1 WebAPI

D.4.2.1.1 mdoc request

The following is an example of a server retrieval mdoc request:

```
{
    "version": "1.0",
    "token": "0w4P4mDP_yxnB4iL4KsYwQ",
    "docRequests": [
        {
            "docType": "org.iso.18013.5.1.mDL",
            "nameSpaces": {
                "org.iso.18013.5.1": {
                    "family_name": true,
                    "document_number": true,
                    "driving_privileges": true,
                    "issue_date": true,
                    "expiry_date": true,
                    "portrait": false
                }
            }
```

```
        }
    ]
}
```

D.4.2.1.2 mdoc response

The following is an example of a server retrieval mdoc response:

```
{
  "version": "1.0",
  "documents":
    ["eyJ0eXAiOiJKV1QiLCJhbGciOiJFUzI1NiIsIng1YyI6WyJNSUlCN3pDQ0FaYWdBd0lCQWdJVUZhcDZiTHFR
T0h0NGROZXNvUy9EYm9IU1BjMHdDZ1lJS29aSXpqMEVBd0l3SXpFVU1CSUdBMVVFQXd3TGRYUnZjR2x6SUdsaFkyRX
hDekFKQmdOVkJBWVRBbFZUTUI0WERUSXdNVEF3TVRBd01EQXdNRm9YRFRJeE1UQXdNVEF3TURBd01GbGdJakVUTUJ
FR0ExVUVBd3dLZFhSdmNHbGhJR3AzY3pFTE1Ba0dBMVVFQmhNQ1ZWTXdXVEFUQmdjcWhrak9QUUlCQmdncWhrak9Q
UU1CQndOQ0FBUzRLYTVzUUtCOHQ2b0JJMjM4bXRkOFdRaTdoRWhsNE1YQ21jU0V5a3hUbzdjNUVVdHRHQnkxRm5yS
1BXQlo4MXFJcXpubzNQdDNyRVhpSUw3cHhHUERvNEdvTUlHbE1CNEdBMVVkRWdRWE1CV0JFMlYDWVcxd2JHVkFaWGh
oYlhCc1pTNWpiMjB3SEFZRFZSMGZCQlV3RXpBUm9BK2dEWUlMWlhoaGJYQnNaUzVqYjIwdQhRWURWUjBPQkJZRUZPR
3RtWFAxUmZxdjhmeW9BcFBUVyswa2ttc3BNQjhQTFVZEl3UVlNQmFBRkZUNkk0T2dUQ2pnMlRCNU1tSElERWlCMH
NBTE1BNEdBMVVkRHdFQi93UUVBd0lIZ0RBVkJnTlZIU1VCQWY4RUN6QUpCZ2NvZ1l4ZEJRRURNQW9HQ0NxR1NNNDlC
QU1DQTBjQU1FUUNJQ1Q5OU5zREsxeGhlWFcyTTNVcmVzNzhOYlVuNGFyRjh6K1RDZUVvWlF3VkFpQjRiL1Uxazg4V0
hEK01sZmxiM0NkSHpQd1RoNmZGYVAycGFMVnZRZHJsZHc9PSJdfQ.eyJkb2N0eXBlIjoib3JnLmlzby4xODAxMy41L
jEubURMIiwibmFtZXNwYWNlcyI6eyJvcmcuaXNvLjE4MDEzLjUuMSI6eyJmYW1pbHlfbmFtZSI6IkRvZSIsImdpdmV
uX25hbWUiOiJKYW5lIiwiaXNzdWVfZGF0ZSI6IjIwMTktMTAtMjAiLCJleHBpcnlfZGF0ZSI6IjIwMjQtMTAtMjAiL
CJkb2N1bWVudF9udW1iZXIiOiIxMjM0NTY3ODkiLCJwb3J0cmFpdCI6Il85al80QUFRU2taSlJnQUJBUUVBa0FDUUF
BRF8yd0JEQUJNTkRoRU9EQk1SRHhFVkZCTVhIVEFmSFJvYUhUb3FMQ01CUlQxSlIwUTlRMEZNVm0xZFRGRm9Va0ZEW
DRKZ2FIRjFlM3g3U2x5R2tJVjNqMjE0ZTNiXzJ3QkRBUlFWRlIwWkhUZ2ZIemgyVDBOUGRuWjJkbloyZG5aMmRuWjJ
kbloyZG5aMmRuWjJkbloyZG5aMmRuWjJkbloyZG5aMmRuWjJkbloyZG5aMmRuWjJkbmJfd0FBUkNBQVlBR1FEQVNJQ
UFoRUJBeEVCXzhRQUd3QUFBd0VBQXdFQUFBQUFBQUFBQUFBQUFBQUFBVUdCQUVDQXdmX3hBQXlFQUFCQXdNREFnVUNBd2t
BQUFBQUFBQUJBZ01FQUFVUkJoSWhFekVVRlZGaGNTSkJCNEdoRmpWQ1VuT1Jzc0h4XzhRQUZRRUJBUUFBQUFBQUFBQ
UFBQUFBQUFBQUFBSF94QUFhRVFFQkFRQURBUUFBQUFBQUFBQUFBQUFBQVVFUklURmhfOW9BREFNQkFBSVJBeEVBUHd
DbHU5NGkyaU1weDlhU3ZIME5BX1VzLXdfM1hucC04LWR3bHlPaDBOcmhSdDM3czhBNXpnZXRLOVI2ZmpMYnVOMGRVd
GJ2U3loUFpLU0FCbjM3VWZoXy01WF9BT3VmOFUwaFhlNE4SW5PUkxmYjNweTJpUW9vT08zZkdBZVBldDFpMUJIdlR
ibXhDbVhXdVZvVWM0SHFEVWxia3pKMV9tdTZkY0VVRUVxTHBCQkJQcGc5X3dCUFd2WFRTMHRNM21NdENfSDlGWks5M
lJ4a0VmT1RUQy1tcjJ0VWwxMFFiYzlLWmE1VzZGWkFIcndEeDg0cDNaN3ZIdkVQeEVmY25hZHEwcTdwTlRlaHVuNVB
jTjJPX3dCWHh0XzdYaG9aaFVxRGRZNVVVb2RRbEs3R2NFaFF6UU43enJDTGJYMHN4MjB6Rl94N1hNQlB0bkJ5YWNYR
zRNVzJDdVZKSkNFanNPU1Q5Z0tnZFZXZU5abGcyWTI0bFNWRmExSEpVY2l2b1Q2bzZZNDhXV2cyZUQxY1lfV21HcG4
5dHlrSWRkdEw2SXF6aEx1N3Y4Y1lQOTZxWXc2SlVkdDlvNWJjU0ZKUHNhaTlZUnBhb3FKREx6Q3JRZ3A2YlRKQXh4a
lBBeC1wNzB5YTFWQWdXcUFwVWQ5S0hXeGJYkFWdDJuYmpKSXBnMzZpdm9zYkRUblE2Nm5GRklUdjI0d09fWTBsamE
4OFJKYVo4dTI5UllUbnI1eGs0X2xybSIzbzFLeEFreDVrZU1qbmFpU29KVVNWQWRobjBySGMzcnJwbTV4MUt1VHMxd
DNrb1huQndlUmdrNC1SU2U5bFhsRmNDNUdhS0p5ejNLSjQtM3Z4ZF9UNnFDbmRqT1B5ckpwLXplU1FseC12MTl6aFh
1MmJjY0FZeGstbEZGRkxKT2prIN1YSnQxd2VnbGVkd1NNOV9zQ0NPUGF0MWkwNUdzd2NVbGFubm5CdFV0UXh4NkFVV
VVDNV9SU2VzNllOeGVpTXU4TGNU1FSNmR4eDg1cDNaclJIczBUb1I5eXNuY3RhdTZqUlJRWWRRNmI4OGVaYzhWME9
rQ01kUGRuUDVpbVZ4dHpGeWhlaXlRU2hYM0hkSjlSUlJUNEowYUlVVUpZY3V6Nm9xVlpETzNnZkhPTTlfdFZQRGl0U
W9yY2RoTzF0c1lBb29vRjB5MF9HdmFFRnhTbW5rY0pjVHp4NkVmY1ZoaWFQU21hM0p1TTk2ZXB2RzFLeGdjZGdjazV
IdFJSU0Nsb29vb1BfMlEiLCJkcml2aW5nX3ByaXZpbGVnZXMiOlt7InZlaGljbGVfY2F0ZWdvcnlfY29kZSI6IkEiL
CJpc3N1ZV9kYXRlIjoiMjAxOC0wOC0wOSIsImV4cGlyeV9kYXRlIjoiMjAyNC0xMC0yMCJ9LHsidmVoaWNsZV9jYXR
lZ29yeV9jb2RlIjoiQiIsImlzc3VlX2RhdGUiOiIyMDE3LTAyLTIzIiwiZXhwaXJ5X2RhdGUiOiIyMDI0LTEwLTIwI
n1dfX0sImlhdCI6MTYwOTg1NTIwMCwiZXhwIjoxNjA5ODU1MzIwfQ.JRjQgYpNthh52j3xQ1f6tkoKRBsF8YwH6NlK
Yg2n_pyayOoQYrRPO0aPBeVJ5lgKBzLumjamuvr3C824R_RYHQ"
  ]
}
```

The contents of the JWTClaimsSet in this example is as follows:

```
{
  "doctype": "org.iso.18013.5.1.mDL",
  "namespaces": {

    "org.iso.18013.5.1": {
      "family_name": "Doe",
      "given_name": "Jane",
      "issue_date": "2019-10-20",
      "expiry_date": "2024-10-20",
      "document_number": "123456789",
      "portrait": "_9j_4AAQSkZJRgABAQEAkACQAAD_2wBDABMNDhEODBMRDxEVFBMXHTAfHRoaHToqLCMw
RT1JR0Q9Q0FMVm1dTFFoUkFDX4JgaHF1e3x7SlyGkIV3j214e3b_2wBDARQVFR0ZHTgfHzh2T0NPdnZ2dnZ2dnZ
2dnZ2dnZ2dnZ2dnZ2dnZ2dnZ2dnZ2dnZ2dnZ2dnZ2dnZ2dnZ2dnZ2dnZ2dnb_wAARCAAYAGQDASIAAhEBAxEB_8QAGw
AAAwEAAwEAAAAAAAAAAAAAAAUGBAECAwf_xAAyEAABAwMDAgUCAwkAAAAAAAABAgMEAAURBhIhEzEUFVFhcSJBB
4GhFjVCUnORssHx_8QAFQEBAQAAAAAAAAAAAAAAAAAAAH_xAAaEQEBAQADAQAAAAAAAAAAAAAAAUERITFh_9oA
```

```
DAMBAAIRAxEAPwClu94i2iMpx9aSvH0NA_Us-w_3Xnp-8-dwlyOh0NrhRt37s8A5zgetK9R6fjLbuN0dUtbvSyh
PZKSABn37Ufh_-5X_AOuf8U0hXeZq8InORLfb3py2iQooOO3fGAePet1i1BHvTbmxCmXWuVoUc4HqDUlbkzJ1_
mu6dcEUEEqLpBBBPpg9_wBPWvXTS0tM3mMtC_H9FZK92RxkEfOTTC-mr2tUl10Qbc9KZa5W6FYAHrwDx84p3Z7vH
vEPxEfcnadq0q7pNTehun5PcN2O_wBXxt_7XhoZhUqDdY5UUodQlG7GcEhQzQN7zrCLbX0sx20zF_x7XMBPtnBya
cXG4MW2CuVJJCEjsOST9gKgdVWeNZlw2Y24lSVFa1HJUcivoT6o6Y48WWg2eD1cY_WmGpn9tykIddtL6IqzhLu7v
8cYP96qYz6JUdt9o5bcSFJPsai9YRpaoqJDLzCrQgp6bTJAxxjPAx-p70ya1VAgWqApUd9KHWyEIbAVt2nbjJIpg
36ivosbDTnQ66nFFITv24wO_Y0lja88RJaZ8u29RYTnr5xk4_lrm-so1KxAkx5keMjnaiSoJUSVAdhn0rHc3rrp
m5x1KuTs1t3koXnBweRgk4-RSe9lXlFcA5GaKJyz3KJ4-3vxd_T6qCndjOPyrJp-zeSQlx-v19zhXu2bccAYxk-
lFFFLJOjk-MXJt1wegledwSM9_sCCOPat1i05GswcUlannnBtUtQxx6AUUUC5_RSes6YNxeiMu8LaCSQR6dxx85p3Z
rRHs0ToR9ysnctau6jRRQYdQ6b88eZc8V0OkCMdPdnP5imVxtzFyhKiyQShX3HdJ9RRRT4J0aIUUJYcuz6oqVZDO3g
fHOM9_tVPDitQorcdhO1tsYAoooF190_GvaEFxSmnkcJcTzx6EfcVhiaPSma3JuM96epvG1Kxgcdgck5HtRRSCloooo
oP_2Q",
      "driving_privileges": [
        {
          "vehicle_category_code": "A",
          "issue_date": "2018-08-09",
          "expiry_date": "2024-10-20"
        },
        {
          "vehicle_category_code": "B",
          "issue_date": "2017-02-23",
          "expiry_date": "2024-10-20"
        }
      ]
    }
  },
  "iat": 1609855200,
  "exp": 1609855320
}
```


D.4.2.2 OIDC

An example of an OIDC workflow is as follows:

Step 1 Configuration

Configuration Request:

```
GET /.well-known/openid-configuration/ HTTP/1.1
Host: utopiadot.gov
```

Configuration Response:


```
{
 HTTP/1.1 200 OK
 Content-Type: application/json
{
   "issuer": "https://utopiadot.gov",
   "jwks_uri": "https://utopiadot.gov/.well-known/jwks.json",
   "authorization_endpoint": "https://utopiadot.gov/connect/authorize",
   "token_endpoint": "https://utopiadot.gov/connect/token",
   "userinfo_endpoint": "https://utopiadot.gov/connect/userinfo",
   "end_session_endpoint": "https://utopiadot.gov/connect/end_session",
   "revocation_endpoint": "https://utopiadot.gov/connect/revocation",
   "introspection_endpoint": "https://utopiadot.gov/connect/introspec",
   "device_authorization_endpoint": "https://utopiadot.gov/connect/deviceauthorization",
   "registration_endpoint": "https://utopiadot.gov/connect/register",
   "frontchannel_logout_supported": true,
   "frontchannel_logout_session_supported": true,
   "backchannel_logout_supported": true,
   "backchannel_logout_session_supported": true,
   "scopes_supported": [
      "org.iso.18013.5.1:family_name",
      "org.iso.18013.5.1:given_name",
      "org.iso.18013.5.1:birth_date",
      "org.iso.18013.5.1:issue_date",
      "org.iso.18013.5.1:expiry_date",
      "org.iso.18013.5.1:issuing_country",
      "org.iso.18013.5.1:issuing_authority",
      "org.iso.18013.5.1:document_number",
      "org.iso.18013.5.1:portrait",
```

```
        "org.iso.18013.5.1:driving_privileges",
        "org.iso.18013.5.1:un_distinguishing_sign",
        "org.iso.18013.5.1:administrative_number",
        "org.iso.18013.5.1:sex",
        "org.iso.18013.5.1:height",
        "org.iso.18013.5.1:weight",
        "org.iso.18013.5.1:eye_color",
        "org.iso.18013.5.1:birth_place",
        "org.iso.18013.5.1:resident_address",
        "org.iso.18013.5.1:portrait_capture_date",
        "org.iso.18013.5.1:age_in_years",
        "org.iso.18013.5.1:age_birth_year",
        "org.iso.18013.5.1:age_over_20",
        "org.iso.18013.5.1:issuing_jurisdiction",
        "org.iso.18013.5.1:nationality",
        "org.iso.18013.5.1:resident_city",
        "org.iso.18013.5.1:resident_state",
        "org.iso.18013.5.1:resident_postal_code",
        "org.iso.18013.5.1:resident_conunty",
        "org.iso.18013.5.1:biometric_template_face"
        "org.iso.18013.5.1:family_name_national_character"
        "org.iso.18013.5.1:given_name_national_character"
        "org.iso.18013.5.1:signature_usual_mark"
        "openid",
    ],
    "claims_supported": [
        "org.iso.18013.5.1:family_name",
        "org.iso.18013.5.1:given_name",
        "org.iso.18013.5.1:birth_date",
        "org.iso.18013.5.1:issue_date",
        "org.iso.18013.5.1:expiry_date",
        "org.iso.18013.5.1:issuing_country",
        "org.iso.18013.5.1:issuing_authority",
        "org.iso.18013.5.1:document_number",
        "org.iso.18013.5.1:portrait",
        "org.iso.18013.5.1:driving_privileges",
        "org.iso.18013.5.1:un_distinguishing_sign",
        "org.iso.18013.5.1:administrative_number",
        "org.iso.18013.5.1:sex",
        "org.iso.18013.5.1:height",
        "org.iso.18013.5.1:weight",
        "org.iso.18013.5.1:eye_color",
        "org.iso.18013.5.1:birth_place",
        "org.iso.18013.5.1:resident_address",
        "org.iso.18013.5.1:portrait_capture_date",
        "org.iso.18013.5.1:age_in_years",
        "org.iso.18013.5.1:age_birth_year",
        "org.iso.18013.5.1:age_over_20",
        "org.iso.18013.5.1:issuing_jurisdiction",
        "org.iso.18013.5.1:nationality",
        "org.iso.18013.5.1:resident_city",
        "org.iso.18013.5.1:resident_state",
        "org.iso.18013.5.1:resident_postal_code",
        "org.iso.18013.5.1:resident_conunty",
        "org.iso.18013.5.1:biometric_template_face",
        "org.iso.18013.5.1:family_name_national_character",
        "org.iso.18013.5.1:given_name_national_character",
        "org.iso.18013.5.1:signature_usual_mark",
        "docType",
        "sub"
      ],
      "grant_types_supported": [
        "authorization_code",
        "client_credentials",
        "refresh_token",
        "implicit",
        "urn:ietf:params:oauth:grant-type:device_code"
       ],
      "response_types_supported": [
        "code",
        "token",
```