Work in progress

This is an early draft of a chapter of a book on the foundations of cryptographic authentication being coauthored by Francisco Corella, Sukhi Chuhan and Veronica Wojnas. Please send comments to the authors.

13. ISO/IEC wallet credentials

The ISO/IEC 18013 series of standards is concerned with the design and international use of ISO-compliant driving licences (IDL). Within that series, ISO/IEC 18013-5, published in September 2021, is more specifically concerned with mobile driving licences (mDL) used in attended mode, while ISO/IEC 18013-7, currently under development, is concerned with the use of mobile driving licences in unattended mode. The forthcoming series ISO/IEC 23220, of which part 1 has already been published, will be more generally concerned with mobile eID systems.

This chapter provides overviews of ISO/IEC 18013-5 in Section 13.1, ISO/IEC 18013-7 in Section 13.2, and ISO/IEC 23220-1 in Section 13.3.

13.1 ISO/IEC 18013-5

ISO/IEC 18013-5[1] is an international standard concerned with the design of a mobile driving licence (mDL) credential carried in a native application running in a mobile device. It defines a protocol for presenting the credential to a verifier in a jurisdiction that may be different than that of the issuing authority of the credential.

It specifies an extensible data model for the credential, an interface between the native application and a reader device operated by an attendant on behalf of the verifier, and an interface that allows the reader device to obtain credential data from the issuing authority over the web by presenting a token issued by the native application. It leaves out of scope the interface between the native application and the issuing authority, and the user experience of the holder of the credential as one activates the application, authenticates to the application, and provides consent for presenting selected data elements from the application to the verifier.

The standard was published in September 2021 after six years of development with broad participation by stakeholders from all continents. It introduces technical innovations, provides novel privacy features, and makes unconventional use of existing technologies. But it is also very complex, and there are vulnerabilities hidden in the complexity. In Sections 13.1.1-8 we provide an overview of the standard, describing the data model and the presentation protocol. In Section 13.1.9 we delineate the security posture of the presentation protocol, describing vulnerabilities to cloning, man-in-the-middle attacks, and unauthorized access, and suggesting mitigations of the vulnerabilities and modifications of the standard that could prevent the attacks. In Section 13.1.10 we discuss the extent to which the presentation protocol supports privacy recommendations made in Annex E of the standard, and in Section 13.1.11 we suggest aspects of a user experience that would meet accessibility requirements and facilitate adoption.

13.1.1 Extensible data model

Table 5 of the standard lists a collection of mandatory and optional data elements that must or may be included in an mDL application. Most of the data elements are attributes, or claims, of the credential holder.

The mandatory elements are: family name, given name(s), date of birth, date of issue, date of expiry, issuing country, issuing authority, licence number, portrait of the mDL holder (a facial image in JPEG or JPEC2000 format), driving privileges (as defined in ISO/IEC 18013-1:2018), and distinguishing sign of the issuing country (as defined by the UN for incorporation into the registration plate, and encoded according to ISO/IEC 18013-1:2018). The values of the family name and given name(s) elements are encoded in the ISO/IEC 8859-1 Latin alphabet No. 1.

Optional elements include: family and given name(s) in UTF-8 national characters; nationality; place of birth; data elements defining the place of residence, including full address; age attestations (discussed below in Section 13.1.4); physical characteristics (height, weight, hair and eye color); image of the handwritten signature or usual mark of the credential holder, date when the portrait was taken; facial, fingerprint, iris or other biometric template(s), and administrative elements. If the optional facial biometric is included, it is encoded in the same jpeg format as the mandatory portrait, and thus would seem to be redundant.

The family and given name(s) in national characters are an acknowledgement of the cultural differences in personal names and the need to accommodate them in government-issued credentials,[2] but not a solution. A solution would have to take into account not only the characters used to write a name but also the parts of a name and the names of those parts (the first, middle and last names, the middle initial, the multiple first names, the hyphenated maiden name, the single very long name used in Madagascar, the father's and mother's surnames used in Spain, etc. Culture-specific data elements would be needed.

Three concepts are used to make the data model extensible: *namespaces*, *documents*, and *device-signed* data elements:

- Each data element belongs to a namespace and has an identifier that is unique within that namespace. The data elements of Table 5 belong to a namespace called "org.iso.18013.5.1", but an mDL issuing authority may define an extension of that namespace to add elements relevant to the authority. The standard gives as examples the United States namespace "org.iso.18013.5.1.US" and the Iowa namespace "org.iso.18013.5.1.US-IA".

- The data elements of Table 5 belong to a document with document type "org.iso.18013.5.1.mDL". But the standard could be adapted for use as a general-purpose digital identity standard by defining other document types, with data elements in namespaces not necessarily related to driving licenses. The standard allows data elements from multiple namespaces to be included in a single document, and multiple documents to be included in an application.

- The data elements in org.iso.18013-5.1.mDL are intended to be certified by a signature computed with the private key of a driving licence issuing authority. And data elements in other documents might be similarly certified by issuing authorities of other kinds of credentials. But general-purpose identity standards also need to accommodate self-asserted user attributes. The standard uses to that purpose data elements that are signed by the private key of the device itself, referring to them as being device-signed rather than issuer-signed.

13.1.2 Terminology

The generality provided by the ability to define arbitrary documents and namespaces led to the introduction of new terminology and acronyms during the development of the standard[a]. A mobile driving license (mDL) became a special case of a mobile document (mdoc), an mDL reader became a special case of an mdoc reader, an mDL application became a special case of an mdoc application, and the term "mdoc application" was abbreviated as "mdoc", which is now defined in Section 3.2 of the standard as referring to either a document or an application.

As a result, the "mobile driving licence (mDL) application" that is the subject matter of ISO/IEC 18013-5 according to its title is now called the "mdoc", and a reader that interfaces with the mDL application is called an "mdoc reader". We shall use this terminology here, while still referring to ISO/IEC 18013-5 as the mDL standard.

13.1.3 Selective disclosure

In traditional third-party cryptographic authentication with a public key certificate, the issuer binds the public key of the subject to a list of certified attributes by applying a signature to the public key and the attributes. The relying party receives the certificate and verifies that all the attributes are certified by validating the signature.

But in the credential presentation protocol specified by the mDL standard, the mdoc reader requests specific data elements and only receives those among the requested elements that the credential holder consents to disclose.

[a] The original terminology can be found at https://mobiledl-e5018.web.app/ISO_18013-5_E_draft.pdf.

The mdoc reader needs to ascertain that each data element that it receives is certified by the credential issuer. One way to do that would be for the issuer to apply a separate signature to each element.

The standard uses instead a more efficient method that only requires one signature. The mdoc has a key pair that it uses to authenticate to the mdoc reader, comprising an authentication public key and an authentication private key. The single signature is applied by the issuer to a data structure called the Mobile Security Element (MSO), comprising the authentication public key of the mdoc and a collection of digests, or cryptographic hashes, one for each certified data element. Each digest is labeled by a digest ID unique within the collection. The digest for a particular data element is computed on its digest ID, the element's identifier, the element's value, and a random value that is generated and associated with the element when the credential is created.

In response to a data retrieval request, the mdoc reader receives the MSO, and the digest ID, element identifier, element value, and random value of each certified data element that is disclosed. The reader verifies the issuer's signature on the MSO using the issuer's public key, which it finds in a "verified issuer certificate authority list" (VICAL). To verify that a disclosed element has been certified, the reader computes the cryptographic hash of its digest ID, element identifier, element value and random value, and compares the result to the corresponding digest in the MSO. If the computed hash matches the digest in the MSO, the collision resistance property of the cryptographic hash function used for digest computation ensures that the data element received by the mdoc reader is identical to the one whose digest is included in the MSO signed by the issuer.

The purpose of the random values associated with the data elements and included in the digest computations is to prevent guessing attacks by the mdoc reader against the certified data elements that are not disclosed, whose digests are also included in the MSO. Without the random values, brute force guessing attacks would easily succeed against undisclosed data elements having relatively few possible values, such as the birthdate data element.

13.1.4 Age attestations

Selective disclosure makes it possible to use age attestation data elements to certify that the credential holder meets a minimum-age requirement without disclosing the birthdate.

The data model includes an age-in-years attestation and a birth-year attestation, and it allows the issuing authority to provision age attestations of the form age-over-nn with TRUE or FALSE values, where nn can be any value from 00 to 99. Section 7.2.5 of the standard specifies a "nearest true attestation above request" algorithm for providing a certified response to a request that asks for an age-over-nn attestation that has not been provisioned. The response provides the age-over-xx data element with value TRUE for the smallest value of xx greater than nn, if any, or the age-over-xx data element with value FALSE for the greatest value of xx less than nn, if any.

Age attestations are a simpler alternative to a method used in the Camenish-Lysyanskaya anonymous credential described in Chapter 3 for certifying that age requirements are met without disclosing the birthdate of the subject.

13.1.5 Session encryption and mdoc authentication

Typical cryptographic communication protocols such as TLS, IPsec or SSH use asymmetric cryptography to authenticate one or both communicating parties by proof of possession of their private keys, then use symmetric cryptography to encrypt and authenticate each record, packet or segment sent between the parties in each direction of traffic, using either separate keys for encryption and message authentication or a single key for authenticated encryption.

The credential presentation protocol of the mDL standard does the opposite.

First, symmetric cryptography is used to set up an encrypted session between the mdoc and the mdoc reader, with authenticated encryption provided by the Advanced Encryption Standard (AES) in Galois Counter Mode (GCM) with a 256-bit symmetric key for each direction of traffic. The symmetric keys are derived from a shared secret established by an ECDH key exchange between the mdoc and the mdoc reader using ephemeral (i.e., single-use) key pairs, with no authentication of either party. The session provides confidentiality and integrity protection for data retrieval requests sent by the mdoc reader, and responses containing data elements returned by the mdoc.

Then the mdoc authenticates to the mdoc reader, possibly multiple times, as follows. For every mdoc response, and every document from which data elements are returned in the response, the mdoc uses its authentication private key to compute a signature on a data structure that comprises the device-signed elements included in the document that are returned in the response. Thus mdoc authentication has a dual purpose: it authenticates the mdoc by proof of possession of the private key, and it authenticates the device-signed data elements in each document within each response.

The structure signed by the mdoc has the generic name “DeviceAuthenticationBytes”, but there is a different instance of the structure for each document. Repeated authentication of the mdoc for each document in each response may seem inefficient, but in the prototypical use case where a mobile driving licence is presented to a traffic control officer, there will be only one response and only one document (org.iso.18013.5.1.mDL).

The standard specifies two different kinds of signature for mdoc authentication, called ECDH-agreed MAC and ECDSA/EdDSA signature. The ECDSA/EdDSA signature is a conventional asymmetric signature. The ECDH-agreed MAC id a symmetric signature, i.e., a message authentication code (MAC), computed with a symmetric key derived from an ECDH shared secret established between the mdoc and the mdoc reader using the authentication key pair of the mdoc and the same ephemeral key pair that the mdoc reader uses to set up the encrypted session.

A privacy benefit of the MAC method, pointed out in Section 8.2.3.3 of the standard, is that the mdoc can repudiate the signature, since it is a symmetric signature that could have been computed by the mdoc reader.

13.1.6 Authentication of the mdoc reader

Like the mdoc, the mdoc reader has an authentication key pair that it can use to authenticate to the mdoc, as described in Section 9.1.4 of the standard. The authentication private key is used to sign each data retrieval request with an ECDSA/EdDSA signature, and the public key is included in a certificate that is sent along with the request.

However, authentication of the mdoc reader is optional, and, furthermore, it is discouraged by the following stipulation in Section 7.2.1: “An mDL shall not require mdoc reader authentication as a precondition for the release of any of the mandatory data elements”. Privacy implications of this stipulation are discussed below in Section 13.1.10.

As we shall see below in Section 13.1.7, the mdoc reader may also authenticate with a TLS client certificate as it retrieves data elements from the issuing authority over the web, but TLS client authentication is also optional.

13.1.7 Transaction flows

As shown in Figure 3 and explained in Section 6.3.2 of the standard, data exchange has an initialization phase, a device engagement phase, and a data retrieval phase.

During the initialization phase the mdoc is activated by the credential holder, or by the mdoc reader using NFC.

Device engagement may take place over an NFC connection between the mdoc and the mdoc reader, or by means of a QR code displayed by the mdoc and scanned by the mdoc reader. Both of these engagement patters imply in-person authentication. Online authentication will be specified in ISO/IEC 18013-7.

During the data retrieval phase, the mdoc reader may perform “device retrieval”, obtaining data elements from the mdoc over BLE, NFC or Wi-Fi after an encrypted session is set up as explained above in Session 13.1.5; or it may perform "server retrieval", obtaining data elements on the web using a Web API over TLS, or OpenID Connect (OIDC).

During device engagement the mdoc transmits to the mdoc reader a device engagement structure defined in Section 8.2.1.1 of the standard, which contains information for setting up device retrieval or server retrieval. The device retrieval information comprises configuration information for BLE, NFC, and Wi-Fi. The server retrieval information includes the URL of an issuing authority server, and a server retrieval token that authorizes the mdoc to retrieve data elements from the server. The mdoc reader has the option to perform server retrieval immediately after device engagement, or to begin by performing device retrieval and switch to server retrieval later, after obtaining fresh server retrieval information as a data element through device retrieval.

In addition to configuration information, the device engagement structure contains the ephemeral public key that the mdoc will use to perform the ECDH key exchange with the mdoc reader and set up the encrypted session if device retrieval is used. Since the mdoc reader obtains the device engagement structure via short-range communication (NFC or presentation by the user of a QR code), the mdoc reader will have to be near the mdoc it the encrypted session is successfully set up. The standard assumes that it will then be implicitly authorized to access mdoc data due to its proximity, as implied by a statement in Table 4 that close-range device engagement with session encryption prevents unauthorized access of mDL data. But proximity is not permission, and security implications of this assumption are discussed below in Section 13.1.9.2.

If the device engagement takes place over NFC, and is followed by device retrieval, the transition to device retrieval is performed using static or negotiated NFC handover. In that case, the device engagement structure is transmitted to the mdoc reader in an auxiliary record of a “Handover Select Message”.

3.1.8 Innovative use of OpenID Connect

Server retrieval using the Web API, specified in Section 8.3.3.2.1 of the standard, is straightforward: the request is sent in the body of an HTTP POST request over TLS and the response is encoded in JSON, with the data elements requested from each document contained in a JSON Web Token (JWT) signed with a JSON Web Signature (JWS).

But server retrieval using OpenID Connect (OIDC) is unconventional.

As stated in the abstract of the OIDC specification, “OpenID Connect 1.0 is a simple identity layer on top of the OAuth 2.0 protocol”. OAuth 2.0 is an authorization or access delegation protocol, originally intended as an alternative to Facebook Connect. And the original version of OpenID, dating back to 2005, may have been the first federated identity protocol. The general paradigm for adding an identity layer to an authorization protocol on the web is as follows: the user of an account at a web site delegates access to their account to a relying party (RP), and the RP accesses the account to obtain identity information about the user.

OAuth 2.0 and the OIDC layer on top of it each have multiple configuration options, but Section 8.3.3.2.2 of the standard precisely defines the options to be used for server retrieval using OIDC:

- “The mdoc reader shall use Authorization Code Flow Grant as specified in OpenID Connect Core 1.0 errata set 1, section 3.1.
- 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 response is redirected to the mdoc according to OAuth 2.0 for Native Apps and shall be according to RFC 8252.”

Here are the authorization code flow steps specified in Section 3.1.1 of OpenID Connect Core 1.0 errata set 1:

1. "Client prepares an Authentication Request containing the desired request parameters.
2. Client sends the request to the Authorization Server.
3. Authorization Server Authenticates the End-User.
4. Authorization Server obtains End-User Consent/Authorization.
5. Authorization Server sends the End-User back to the Client with an Authorization Code.
6. Client requests a response using the Authorization Code at the Token Endpoint.
7. Client receives a response that contains an ID Token and Access Token in the response body.
8. Client validates the ID token and retrieves the End-User's Subject Identifier."

Here is how the login_hint parameter is defined in Section 3.1.2.1 of OpenID Connect Core 1.0 errata set 1:

- "OPTIONAL. Hint to the Authorization Server about the login identifier the End-User might use to log in (if necessary) ... The use of this parameter is left to the OP's discretion."

RFC 8252 is concerned with the use of OAuth 2.0 by a client that is a native app. It recommends the use of an external browser, rather than a web view, for communication between the app and the authorization server, and proposes mitigations to the vulnerability caused by the fact that a malicious app may be able to register the same custom scheme as the client app with the operating system of the mobile device.

Putting all this together, OIDC should be used as follows:

1. The native app redirects the default browser to the Authorization Server
2. The Authorization Server Authenticates the End-User.
3. The Authorization Server obtains End-User Consent.
4. The Authorization Server redirects the default browser to the native app with an Authorization Code.
5. The native app requests a response using the Authorization Code at the Token Endpoint.
6. The native app receives a response that contains an ID Token and Access Token in the response body.
7. The native app validates the ID token and retrieves the End-User's Subject Identifier.

But that is not how OIDC is used in the mDL standard.

It cannot be used that way because, in the prototypical use case where a police officer makes a traffic stop and asks for the driving licence:

- The native app is the mdoc reader.
- The browser is the default browser of the mobile device carried by the police officer that hosts the mdoc reader.
- The end-user is the police officer.
- The driver is not involved in the communication between the mdoc reader and the issuing authority.

Thus, it seems at first glance that OIDC is not applicable in this use case. But the standard has made the following unconventional adaptation, as can be seen in the example of Section D.4.2.2 of Annex D:

- The End-User is not authenticated or asked to provide consent;
- The OIDC login_hint parameter is not used as a hint. It is used as an authorization token, which is swapped for the OAuth authorization code;
- The identifiers of the data elements that the mdoc reader requests are listed in the value of the scope parameter of the OAuth authorization request; and
- The values of the data elements are included in the claim set of the ID Token.

13.1.9 Security posture

The credential presentation protocol specified by the standard is vulnerable to cloning and man-in-the-middle attacks due to the existence of an unsafe transaction flow where the mdoc does not authenticate. This vulnerability is discussed below in Section 13.1.9.1. Mitigation of the vulnerability and a suggested modification of the protocol to prevent the attacks are discussed in Section 13.1.9.1.1.

The protocol is also vulnerable to unauthorized access due to its reliance on proximity for implicit mdoc reader authentication. This is discussed below in Sections 13.1.9.2.1-3. Mitigation of the vulnerability and suggested modifications of the protocol to prevent unauthorized access are discussed in Section 13.1.9.2.4.

The cloning and MITM vulnerabilities may result in impersonation of the credential holder if the holder authenticates to the verifier with proof of possession of the credential as the only authentication factor. However, if the credential is the mobile driving licence, the mdoc reader may defeat impersonation by showing the holder's portrait to the attendant.

13.1.9.1 Vulnerability to cloning and man-in-the-middle attacks

The standard states in Section 9.1.3.1 that "the security objective of mdoc authentication is to prevent cloning of the mdoc and to mitigate man in the middle attacks". But there is an unsafe transaction flow where the mdoc does not authenticate.

As shown in Figure 3 of the standard, the protocol has a transaction flow on the left of the figure where the mdoc reader performs device retrieval, and a flow on the right where it performs server retrieval. But the protocol is complicated by the fact that the flow on the right can branch out from the flow on the left in two places, at the top, after the mdoc reader obtains the server retrieval information from the device engagement structure, or further down after it makes a device retrieval request for the information.

As we saw in Section 13.1.5, the mdoc authenticates by sending one or more signatures (ECDH-agreed MACs or ECDSA/EdDSA asymmetric signatures) in each device retrieval response. Therefore, when the server retrieval flow branches out further down, the mdoc authenticates as it provides the server retrieval information to the mdoc reader.

On the other hand, when the server retrieval flow branches out at the top, no device retrieval takes place, and the mdoc does not authenticate. That transaction flow can be exploited for cloning and man-in-the-middle (MITM) attacks as follows.

A "clone" of the mdoc can be defined as a native application implemented by an attacker and running on a mobile device that can behave like the legitimate mdoc when interrogated by and mdoc reader and provide all the data elements requested by the reader.

Such a clone can be created by obtaining the device engagement structure from the legitimate mdoc, storing it in an attack application, and programming the attack application to provide it to the mdoc reader during device engagement, either in the QR code or in the auxiliary data record of the NFC Handover Select Message. The mdoc reader will not be able to perform an ECDH key agreement with the attack application using the ephemeral public key of the legitimate mdoc included in the device engagement structure. But it may not try to do so. Instead, it may use the server retrieval token also included in the device engagement structure to obtain any data elements it wants from the issuing authority. The verifier operating the mdoc reader will then be satisfied and may think that the attack application is the legitimate mdoc, and the person presenting the attack application is the legitimate holder of the credential.

It might be argued that the attack application is not a true clone because it cannot provide data elements via device retrieval, and the mdoc reader would discover the attack if it attempted device retrieval. But the attack application can actually prevent the mdoc reader from performing device retrieval by (i) using the QR code to provide the device engagement structure, which rules out subsequent use of NFC for device retrieval as pointed out in NOTE 1 of Section 6.3.2.5 of the standard; (ii) not supporting Wi-Fi transmission, which is optional, for data retrieval; and (iii) claiming to support BLE but turning it off in the mobile device. The mdoc reader may then fall back on server retrieval without detecting the attack.

A MITM attack can be similarly mounted using an attack application that obtains the device engagement structure from the legitimate mdoc and relays it to the mdoc reader during the attack, instead of obtaining and storing it when building the clone.

13.1.9.1.1 Mitigation and attack prevention

When the mdoc reader of the verifier does not use the unsafe transaction flow, mdoc authentication does meet the objective of preventing cloning and mitigating MITM attacks, because the mdoc reader must in that case make device retrieval requests to obtain data elements or server retrieval information, and will detect cloning or MITM attacks by not being able to verify the signatures in the responses.

When attempting a MITM attack, the attacker will relay requests from the mdoc reader of the verifier to the legitimate mdoc, and responses from the legitimate mdoc to the mdoc reader of the verifier. But the reader will not be able to verify the signatures in the responses, for the following reason. As seen above in Section 13.1.5, an mdoc authentication signature is applied to a DeviceAuthenticationBytes structure. As defined in Section 9.1.3.4 of the standard, the DeviceAuthenticationBytes structure comprises a SessionTranscript structure (which, contrary to what its name suggests, is not a transcript of the encrypted session or any portion thereof). As defined in Section 9.1.5.1, the SessionTranscripst structure comprises among other data, an EReaderKeyBytes structure. As defined in Section 9.1.1.4, the EReaderKeyBytes structure is the ephemeral ECDH public key that the mdoc reader uses to set up the encrypted session. Since the signature will be computed by the legitimate mdoc, the EReaderKeyBytes structure used in the computation will be the ephemeral public key of the reader side of the attacker who is relaying the responses, while the mdoc reader of the verifier will expect it to be its own ephemeral public key.

In a cloning attack, the attacker will make device retrieval requests for data elements and for the server retrieval information to the legitimate mdoc, and save the responses in the clone, to be used when the mdoc reader of the verifier makes the corresponding requests. Assuming that the attacker is not able to forge mdoc authentication signatures of either kind, the signatures in the saved responses will have to be those received from the legitimate mdoc, and verification will fail Because the ephemeral key in the signed data will be the one used by the attacker to establish the encrypted session when constructing the clone, rather than the one used by the mdoc reader when retrieving data from the clone.

Therefore, the cloning and MITM vulnerabilities can be mitigated in the current protocol by not using the unsafe flow. And the attacks could be prevented in a future version of the protocol by not including the server retrieval information in the device engagement structure and thus eliminating the unsafe flow from the protocol.

13.1.9.2 Unauthorized access attacks, mitigations, and prevention

We saw in Sections 13.1.6-7 that cryptographic authentication of the mdoc reader is optional and the standard relies instead on its proximity to the mdoc for implicit authentication. This enables the following unauthorized access attacks.

13.1.9.2.1 Active attack against NFC activation

Section 6.3.2.2 of the standard allows activation of the mdoc to be triggered by an mdoc reader using NFC. NFC communication can take place at a distance of up to 10 cm. In crowded public transportation, a rogue mdoc reader held by an attacker could come within a few centimeters of an mdoc carried by a credential holder and activate the mdoc.

Furthermore, 10 cm is a nominal distance for communication between commercial off-the-shelf devices. Much longer distances can be achieved with special antennas. An active attack from a distance of 50 m, implementable with a loop antenna built into a briefcase, has been reported[3].

After activating the mdoc, the rogue mdoc reader could perform a static or negotiated NFC handover with the mdoc and obtain the device engagement structure containing the server retrieval token and URL from the mdoc in the auxiliary record of the Handover Select Message. It could then retrieve data elements from the issuing authority without authentication.

This active attack can be prevented by configuring the mdoc to require activation by the holder. But there are also passive attacks that may be more difficult to mitigate.

13.1.9.2.2 Eavesdropping attacks on NFC device engagement

A common use of a mobile driving licence will no doubt be to provide proof of age for entering a bar. The selective disclosure feature of the standard will allow a bouncer to use an mdoc reader to obtain over-18 age attestations from customers carrying mobile driving licences as they enter the bar, without obtaining any other information about the customers. An attacker seeking information about a particular customer may wait near the door for the customer to enter and use a reader with a special antenna to eavesdrop on the NFC communication between the customer and the bouncer. Eavesdropping on NFC from distances of several meters has been reported by NIST researchers[4]. Although the bouncer will only retrieve the age attestation, the attacker may extract the server retrieval token and URL from the device engagement structure and use them to retrieve other data elements from the issuing authority, as in the above active attack.

A mitigation for this attack could be based on disallowing data retrieval from two different mdoc readers at about the same time. Since the mdoc would not be able tell which reader is malicious, it would disallow both. To that purpose it would wait a few seconds before granting a data retrieval request and disallow both requests if a second one arrived.

13.1.9.2.3 Eavesdropping attacks on the QR code

Instead of NFC, the bouncer in the above example could use the QR code to perform device engagement. QR codes are designed for a 10:1 distance-to-size ratio, so a 2 cm wide QR code should be scanned from a distance of 20 cm. However, attack software in the eavesdropping device could take a very high resolution picture of the QR code from a distance of several meters and enlarge it before scanning it. The attacker could then use the device engagement structure in the QR code to obtain the server retrieval token and URL and use them to retrieve data elements from the issuing authority as in the NFC version of the attack.

The same mitigation as in the NFC version could be used.

13.1.9.2.4 Prevention of unauthorized access attacks

In a future version of the standard, unauthorized access could be prevented by requiring the mdoc reader to authenticate when making device retrieval requests, and not including the server retrieval token in the device engagement structure.

An objection to this change would be the stipulation in Section 7.2 of the standard that "An mDL shall not require mdoc reader authentication as a precondition for the release of any of the mandatory data elements". The motivation for this stipulation may be to facilitate the use of the mDL in a different jurisdiction than that of the issuer, where mDL readers may present certificates that the mDL may not be able to verify. If that is indeed the motivation, the objection could be overcome by allowing the mdoc holder to allow unauthenticated access in a different jurisdiction, or more generally as needed.

13.1.10 Privacy

Many aspects of the credential presentation protocol that have an impact on the holder's privacy are outside the scope of the standard, such as the interface between the mdoc and the issuing authority, how the credential is provisioned to the mdoc, the information in the server retrieval token, and the user experience of the credential holder, including how the holder interacts with the mdoc and with the issuing authority and how the holder chooses the data elements to be disclosed and provides consent to disclose them. The privacy posture of the protocol is therefore undefined.

However, Annex E of the standard includes a list of privacy desiderata, with recommendations on how various parties could help fulfill them. The protocol as specified in Sections 6-9 of the standard supports some of the desiderata, but not others.

One of the privacy desiderata is data minimization. This is supported by the selective disclosure mechanism described above in Section 13.1.3. It is also supported by the age attestation data elements and the algorithm that provides the “nearest true attestation above request” described above in Section 13.1.4. Section E.11.1 of Annex E states that "provisioning ten age verification statements of age greater than 15 through 21, and 25, the current age and one more than the current age, mDL verifiers should be able to confirm age around the globe without revealing Date of Birth."

Another desideratum is informed consent, which requires knowledge of the identity of the verifier by the credential holder, and hence authentication of the mdoc reader. This is hindered by the lack of authentication of the mdoc reader in device retrieval, and the stipulation in Section 7.2.1 of the standard that "An mDL shall not require mdoc reader authentication as a precondition for the release of any of the mandatory data elements". It is also hindered by the fact that TLS client authentication is optional. TLS client authentication during server retrieval could convey the identity of the verifier to the issuing authority, which could then convey it to the credential holder via the out-of-scope Interface 1 of Figure 1.

Yet another privacy desideratum is transaction unlinkability. This is hindered by the fact that the mdoc authentication public key included in the MSO could be used to track the holder. Section E.8.4 of Annex E recommends using key rotation as a mitigation. However, key rotation would not prevent transaction linking by the issuing authority in collusion with the verifiers.

A privacy feature that is not on the list of desiderata is unobservability of transactions by the issuing authority. Device retrieval provides this feature, but server retrieval does not, as pointed out in NOTE 2 of Section 8.3.2.2 of the standard, which states that:

"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."

The standard does not explain what those mitigating strategies could be and how they could be communicated to the holder.

A novel privacy feature introduced by the standard is the inclusion in both device retrieval and server retrieval requests of an indication, for each data element being requested, of whether the verifier intends to retain the element.

13.1.11 User experience

13.1.11.1 Device activation and engagement

13.1.11.2 Holder authentication

13.1.11.3 Attribute selection and holder consent

13.1.11.4 Accessibility

13.2 ISO/IEC 18013-7

13.3 ISO/IEC 23220-1

[1] ISO/IEC 18013-5:2021: Personal identification — ISO-compliant driving licence — Part 5: Mobile driving licence (mDL) application. https://www.iso.org/standard/69084.html.

[2] Government of Canada. Reclaiming Indigenous names on Immigration, Refugees and Citizenship Canada identity documents. https://www.canada.ca/en/immigration-refugees-citizenship/news/2021/06/reclaiming-indigenous-names-on-immigration-refugees-and-citizenship-canada-identity-documents.html

[3] G.P. Hancke. A practical relay attack on ISO 14443 proximity cards. http://www.rfidblog.org.uk/hancke-rfidrelay.pdf.

[4] D. R. Novotny, J. R. Guerrieri, M. Francis and K. Remley, "HF RFID electromagnetic emissions and performance," 2008 IEEE International Symposium on Electromagnetic Compatibility, Detroit, MI, USA, 2008, pp. 1-7, doi: 10.1109/ISEMC.2008.4652133.