ETSI TR 119 476 V1.1.1 (2023-08)

Electronic Signatures and Infrastructures (ESI);
Analysis of selective disclosure and zero-knowledge proofs applied to Electronic Attestation of Attributes

Reference
DTR/ESI-0019476

Keywords
identity, trust services

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Technical Report (TR) has been produced by ETSI Technical Committee Electronic Signatures and Infrastructures (ESI).

Modal verbs terminology

In the present document "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Executive summary

The eIDAS2 regulation and the Architecture Reference Framework (ARF) define regulatory requirements on selective disclosure and unlinkability for the EUDI Wallet.

The present document provides a comprehensive analysis of signature schemes, credential formats and protocols that cater for selective disclosure, unlinkability, predicates and Zero-Knowledge Proofs.

More specifically, the present document analyses the ISO mobile driving license (ISO mDL) and W3C Verifiable Credentials (VCs) in conjunction with SD-JWT, since these credentials formats have been selected as Person Identification Data (PID) and Qualified Electronic Attestation of Attributes (QEAAs) for the EUDI Wallet.

The ISO mDL and SD-JWT formats and related presentment protocols cater for selective disclosure and unlinkability based on the concept of hashes of salted attributes. Furthermore, these credential formats support SOG-IS approved cryptographic algorithms and can also be used with quantum-safe cryptography for future use.

The present document gives recommendations on how eIDAS2 compliant QTSPs should issue PID/(Q)EAAs in the form of ISO mDL and/or SD-JWT that cater for selective disclosure and unlinkability. Furthermore, there are recommendations on how to store such credential formats in the EUDI Wallet, and how to present selectively disclosed attributes to eIDAS2 relying parties.

Introduction

A historical perspective

To facilitate an understanding of the concepts in the present document, the present clause begins with a brief account of the history of selective disclosure and Zero-Knowledge Proofs, the problems they were introduced to address, their applications, and their potential uses in electronic attestations of attributes. The present document also discusses related concepts where required.

Cryptographic schemes for selective disclosure, unlinkability, blinded signatures, Zero-Knowledge Proofs (ZKP), predicates and range proofs have been researched and developed since the 1980s. The first Zero-Knowledge Proof scheme was published in a paper 1985 [i.62] by the researchers Shafi Goldwasser, Silvio Micali, and Charles Rakoff. The abstract of this paper defines ZKP as: "*Zero-Knowledge Proofs are defined as those proofs that convey no additional knowledge other than the correctness of the proposition to the question*".

The present document on selective disclosure can be linked to the broader work on signatures that allow for updates to the signed document. In their 1994 paper "Incremental Cryptography: The Case of Hashing and Signing" [i.5], Bellare, Goldreich, and Goldwasser investigate cryptographic transformations where the updates to the results are proportional to the amount of modification done. Using digital signatures as a case, the authors propose the idea of updating the signature upon modification of the underlying message in a way that is proportional to the amount of change in the message (as opposed to simply signing the new message). The authors called for future work to explore various operations, such as delete and update, that could be supported by incremental signatures.

It is important to note that ZKP is not a selective disclosure scheme in and of itself, but rather a property of a proof system. Goldwasser, Micali, and Rakoff (1985) defined ZKP [i.62] as "*those proofs that convey no additional knowledge other than the correctness of the proposition to the question*". Thus, ZKP is not limited to selective disclosures or signatures proofs in the context of electronic attestations of attributes. On the contrary, Brassard et al. demonstrated in their paper "*Minimum disclosure proofs of knowledge*" [i.17] that everything that has a proof, also has a ZKP version of that proof.

Put differently, every selective disclosure related proof has a ZKP version of that proof. But it is incorrect to state that every selective disclosure scheme is done using ZKP, or that every ZKP is used for selective disclosure.

Electronic attestations of attributes represent a context in which several features, such as selective disclosure or proofs about knowledge of states like a valid signature value, have been implemented with the ZKP property. Among the earliest work here was done by Feige, Fiat, and Shamir (1987) who demonstrated how ZKP can be used in identification schemes by a user demonstrating knowledge as opposed to prove the validity of assertions. Since then, ZKP has been widely deployed in many of the privacy focused selective disclosure capable electronic attestation of attribute solutions.

Another pioneer in the field of ZKP was the American cryptographer David Chaum who published the scientific paper Blind Signatures for Untraceable Payments [i.30] in 1982, which described anonymized digital money (DigiCash) for the first time. The concept of Blind Signatures was designed to ensure complete privacy of users who wanted to conduct online transactions.

In 2002, Steinfeld, Bull, and Zheng published their paper "Content Extraction Signatures" (CES) [i.117]. In it, the authors present a way to perform the delete operation without knowledge of the signer's private key. The authors argue that this would allow a user "to disclose only certain parts of a document" as opposed to "forcing the document holder to disclose all of its contents to a third party for the signature to be verifiable". The authors then go on to present the idea of context extraction, i.e. "the extraction of certain selected portions of a signed document" in cases where a user "does not wish to pass on the whole document to a third (verifying) party". Their method is based on signing digests of data subsets. Relatedly, Johnson et al., (2002) presented their work on redactable signatures, which are conceptually very similar to CES. In fact, the proposed schemes in the papers overlap, together detailing four different schemes for CES. Two of these rely on commitment vectors, and two on the homomorphic properties and batching of RSA respectively.

Brands (2002) directly applies these concepts to electronic attestations of attributes. In his 2002 paper "A Technical Overview of Digital Credentials" [i.15] Brands discusses the "selective disclosure properties of data fields" in Digital Credentials. In that paper, Brands presents the idea to "hash attributes [...] using a collision-intractable hash function; to disclose these attributes, Alice discloses the preimages of the corresponding [attributes]". Interestingly, Brands proposed design also relies on a proof of knowledge of the digital signature, which is among the first references to the use of ZKP for enhancing privacy when presenting electronic attestations of attributes. Brands' paper is also among the earliest work on the use of predicates in electronic attestations of attributes. In essence, Brands' work was based on commitment vectors and the algebraic manipulations of these, allowing proofs containing AND, OR, and NOT connectives between attributes and for a single attribute.

The above mentioned work laid the groundwork for the concept of selective disclosure. Ongoing work presented workarounds to discovered vulnerabilities in some of the proposed schemes, and introduced more advanced features that further improved privacy e.g. by enabling multi-show unlinkable selective disclosures (see "Anonymous Credentials" [i.24] by Camenisch and Lysyanskaya in 2003). Notable early examples of implementations of this work focused on enhanced privacy include AnonCreds and Idemix (both based on Camenisch-Lysyanskaya signatures as detailed herein under clause 4), as well as U-Prove (based on Brands' work). A more recent example of a multi-message signature scheme capable of selective disclosure is the BBS signature scheme (detailed in clause 4.3 and is based on group signatures and the work of Boneh, Boyen, and Shacham, 2004). However, as noted in Camenisch et al. (2013) [i.24], real-world deployments of cryptographic primitives, schemes and protocols in electronic attestations of attributes have been slow due to them being hard to understand and "very difficult to use" as they often require advanced cryptography and the combination of several protocols to achieve the desired privacy goals. In a survey, Ashgar (2011) [i.4] lists some of these often employed mechanisms, including blind signatures (Chaum, 1983), zero-knowledge Proofs (ZKP) (Goldwasser, Micali, and Rakoff, 1985), group signatures, commitment schemes (formalized in Brassard, Chaum, and Crépeau, 1988 [i.17]), and multi-message signing; which often need to be employed in tandem to reach privacy goals important for selective disclosure including multi-show unlinkability, blinding, and the ability to present a subset of the signed attestation.

In contrast to the focus on increasing privacy, others sought more performant schemes with lower but still acceptable levels of privacy. A notable example here is the early work of Bull, Stanski, and Squire (2003) [i.19], who presented a way to "enable selective disclosure of verifiable content" using a randomized salt to blind the attribute disclosures, using an identifier for each disclosable attribute, and the principle of signing the hash digests of attributes. To disclose the desired attributes, a user would simply present a subset of the attestation to the verifier, together with the attributes and salts to disclose. Variations of this salted hash digest based approach is used both in the ISO/IEC 18013-5:2001 [i.87] standard and in the IEFT SD-JWT specifications. Note that these techniques do not achieve the same levels of privacy as their more advanced counterparts (e.g. U-Prove, AnonCreds, Idemix, and BBS), but they are easier to use and more performant.

The academic research of cryptographic schemes for selective disclosure, ZKP, and predicates have continued from the mid 2010s until present day: Bulletproofs [i.19] and Pointcheval-Sanders Multi-Signatures [i.110] provide range proofs over committed values, whilst zk-SNARK (clause 4.5.3) and zk-STARK (clause 4.5.4) are advanced protocols for ZKP. More information about those cryptographic schemes are described in clause 4 of the present document.

The Internet standardization organizations Hyperledger, IETF and W3C® have followed the academic cryptographic research by creating Internet standards for selective disclosure, ZKP and predicates. Hyperledger has specified AnonCreds [i.64]. IETF has specified the BBS Signature Scheme [i.71], JSON WebProofs [i.74], PKIX attribute certificates [i.78], and SD-JWT [i.76]. W3C has specified BBS Cryptosuite and the Verifiable Credentials Data Model describes ZKP [i.128]. Furthermore, ISO/IEC 18013-5 [i.87] specifies selective disclosure for the mobile driving license by introducing the Mobile Security Object (MSO) for the device retrieval use case. Clauses 5 and 6 in the present document describe the mentioned standards in more detail.

Overview and use cases

An overview of various use cases is provided in Figure 1 to illustrate the concepts of selective disclosure, unlinkability, ZKP, predicates and range proofs.

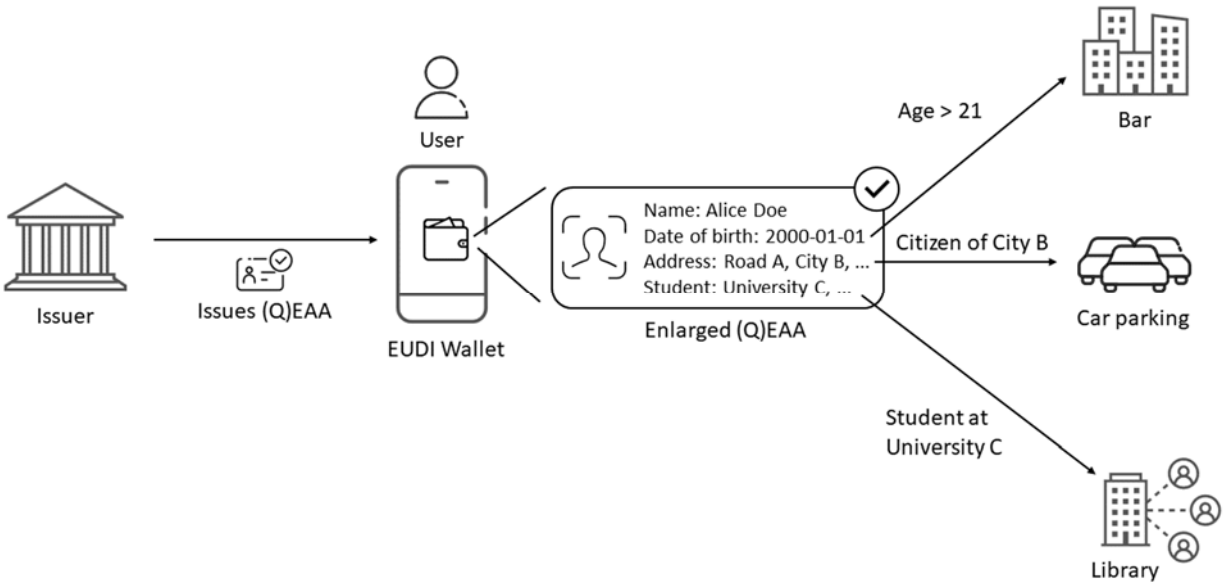

Figure 1: Overview of selective disclosure

First, an issuer creates and issues a (Qualified) Electronic Attestation of Attribute (EAA) (denoted as (Q)EAA) to a user, whereupon the (Q)EAA is stored in the user's EUDI Wallet.

EXAMPLE 1: The (Q)EAA contains the attributes name (first name and last name), date of birth, address (street, city, zip code, etc.), and student information (university, exams, course, etc.).

NOTE 1: The issuer may also issue a Person Identification Data (PID) with the same attributes, but a (Q)EAA is used for readability in this particular example.

The (Q)EAA that is stored in the user's EUDI Wallet is also associated with cryptographic keys that are necessary for the cryptographic scheme's selective disclosure capabilities. In order to access the private keys, the user needs to authenticate with PIN-code or biometrics. Sections 6.3 and 6.5.3 in the Architecture Reference Framework v1.0.0 [i.34] provide more information on the EUDI Wallet security architecture and the supported cryptographic keys management systems.

Now, the user can use its EUDI Wallet to present selected attributes of the (Q)EAA to various relying parties. A user may present multiple attributes to each verifier and is not limited to present only a single attribute claim. The user may also be able to create a presentation that includes claims from at least two (Q)EAAs even if these are issued by different issuers (herein referred to as combined presentation).

When going to a bar, for example, the user may only present a proof that she is over the age of 21 years.

NOTE 2: This is an example of a selective disclosure that can be performed as a predicate proof. The EUDI Wallet contains the user's actual date of birth (2000-01-01), but the EUDI Wallet only presents a proof that $21 \leq age$.

NOTE 3: This example can also be achieved using selective disclosure of a single attribute. The EUDI Wallet could contain an attestation with the key value pair `"age_over_21": "True"`.

When parking the car in City B, the user may present a proof that she is a citizen of City B in order to get a discount when paying for the parking ticket. Unlinkability here prevents behavioural profiling and the user presents only a proof of knowledge of the undisclosed issuer's signature (the signature is linkable data).

NOTE 4: This can be achieved using a ZKP. The EUDI Wallet only presents a ZKP of knowledge of a valid signature without disclosing said signature.

When borrowing a book a the university library, the user may only present that she is taking Course D at University C to prove that she is eligible to borrow the course literature.

NOTE 5: This is an example of selective disclosure of a single attribute. The EUDI Wallet contains detailed student information (university, degrees, courses, etc.), but the EUDI Wallet only presents the single claim that user studies at University C.

The concept of verifier unlinkability relates to the amount of additional information that colluding verifiers can discover about the user. High unlinkability means that the colluding verifiers learn little in addition to what the user disclosed to each verifier. Similarly, a single verifier cannot collect multiple selectively disclosed attributes and link them to the same user. This requires removing correlatable data in the presentation to each verifier.

EXAMPLE 2: If presentations are unlinkable, then the bar (who knows that the user is over 21 years) cannot cooperate with the car parking (who knows that the user lives in City B) to link the user's age to the citizenship.

EXAMPLE 3: If presentations are unlinkable, then the user may visit the university library multiple times and present proofs of different courses (Course D, Course E, etc.) over time. The university library cannot link the different courses to the same user.

The concept of issuer unlinkability means that the issuer cannot collude with one or more verifiers to discover where the user is using the issued (Q)EAA.

Selective disclosure in eIDAS2

The proposed regulation amending Regulation (EU) No 910/2014 (commonly called eIDAS2) [i.54] defines the term selective disclosure as follows in recital 29:

"The European Digital Identity Wallet should technically enable the selective disclosure of attributes to relying parties. This feature should become a basic design feature thereby reinforcing convenience and personal data protection including minimization of processing of personal data."

The ARF [i.34] also defines the term selective disclosure as follows in section 2:

"The capability of the EUDI Wallet that enables the User to present a subset of attributes provided by the PID and/or (Q)EAAs."

Furthermore, in the ARF outline [i.33] the term unlinkability is also introduced as follows in section 5:

"The Wallet shall ensure an appropriate level of privacy, implementing policies about non-traceability and unlinkability of user's activities for third parties as appropriate considering:

- *the applicable legal context for identity providers and attestation providers;*
- *the need to retain evidence for dispute resolution purpose;*
- *the right for the user to be informed of the use of their EUDI Wallet."*

More specifically, the ARF [i.34] mandates ISO/IEC 18013-5 [i.87] Mobile Security Object (MSO) and IETF SD-JWT enable selective disclosure of attributes from a (Q)EAA or the PID contained in the EUDI Wallet. The ARF allows for other selective disclosure techniques based on multi-message signature schemes.

Selective disclosure capabilities

The proposed amendments in eIDAS2 [i.54] offer limited guidance on how to interpret the requirement of selective disclosure. For instance, it is not specified whether the EUDI Wallet:

- enables selective disclosure of attributes from only a single attestation or if the disclosed attributes can originate from multiple different attestations;
- requires that the disclosures originate from attributes that are issued by the same issuer;
- is capable of multi show unlinkable disclosures in order to prevent a verifier to learn more about the user with each subsequent presentation;
- prevents improper pairing of disclosable attributes.

To clarify the requirement of selective disclosure it is helpful to list possible requirements. Hence, Table 1 lists descriptions of common selective disclosure requirements for various schemes.

Table 1: Common selective disclosure requirements

Req	Common selective disclosure requirements
1	The possibility that the user selectively discloses attributes so that these attributes appear to be part of an attestation other than the one they were originally part of.
2	The possibility to selectively disclose attributes from at least two separately issued attestations issued by the same issuer.
3	The possibility to selectively disclose attributes from at least two separately issued attestations issued by different issuers.
4	The possibility to selectively disclose attributes from a single attestation.
5	The selectively disclosed attributes are unlinkable by means other than the information shared in the attribute, over multiple sharing sessions of the disclosed attributes to at least two different verifiers who can collude and compare the attribute disclosures they have received.
6	The selectively disclosed attributes are unlinkable by means other than the information shared in the attribute, over multiple sharing sessions of the disclosed attributes to the same verifier.
7	The selectively disclosed attributes are unlinkable by means other than the information shared in the attribute, over multiple sharing sessions of the disclosed attributes to at least one verifier who can collude with the (Q)EAA issuer and show the attribute disclosures they have received.
8	Whether or not, the verifier upon receipt of selectively disclosed attributes, is able to confirm that the attributes were issued to the same identity subject that is presenting the attributes (or to an authorized representative thereof) and to no-one else.
9	Whether or not, the verifier upon receipt of selectively disclosed attributes, is able to confirm that the attributes describe the same identity subject that is presenting the attributes (or to an authorized representative thereof) and to no-one else.

The above requirements do not represent a complete list but are only common grounds for comparison between different selective disclosure mechanisms. It should also be noted that not all schemes satisfy all requirements. A condensed list that can be used to assess the different selective disclosure mechanisms could be limited to:

1) Does the mechanism support multi-show unlinkability (combination of requirements 5, 6 and 7)?

2) Does the mechanism support combined presentations (combination of requirements 2 and 3)?

3) How resilient is the mechanism against colluding parties (combination of requirements 5 and 7)?

4) Does the mechanism assure holder binding and that the presented disclosures are properly paired (combination of requirements 1, 4 ,8 and 9)?

1 Scope

The present document analyses cryptographic schemes for selective disclosure and their potential application for privacy of electronic attestation attributes in line with the expected requirement of the proposed regulation amending Regulation (EU) No 910/2014 (commonly called eIDAS2) [i.54].

NOTE 1: The term selective disclosure is a collective term that may also include various concepts of unlinkability, Zero-Knowledge Proofs, predicates and range proofs depending on the context of the specific cryptographic scheme. The scope of the present document is primarily to describe selective disclosure and unlinkability of each analysed cryptographic scheme.

NOTE 2: Zero-knowledge proofs, range proofs, and predicates are out of scope in the ARF [i.34]. If an analysed cryptographic scheme relies on any of these features, they will be described in the context of that particular cryptographic scheme.

The present document aims at providing a comprehensive overview of existing cryptographic schemes for selective disclosure and the credential formats and protocols associated with these cryptographic schemes.

The aim of the present document is first to provide input to ETSI standardization relating to how selective disclosure may be applied to the eIDAS2 credentials (Qualified) Electronic Attribute Attestations ((Q)EAA) and Person Identification Data (PID). More specifically, the present report may serve as input to (Q)EAA issuance policies as being specified in ETSI TS 119 471 [i.49] and (Q)EAA profiles as being specified in ETSI TS 119 472 [i.50].

Second, the present document will also analyse the policy requirements for (Q)TSPs and PID providers issuing (Q)EAAs or PIDs with selective disclosure capabilities to EUDI Wallets.

Third, the present document analyses how the user of an EUDI Wallet can present selected attributes of a (Q)EAA or PID to relying parties (or (Q)TSPs acting as relying parties). Consequently, the present document can highlight needs that may require future standardization efforts.

The present document analyses the concepts of selective disclosure, unlinkability, ZKP, predicates, and range proofs in the following main clauses:

- Selective disclosure signature schemes (clause 4): This clause describes the academic research of the cryptographic algorithms and schemes that shape the foundation for selective disclosure signature schemes.
- Selective disclosure credential formats (clause 5): This clause describes the credential formats that have been developed and standardized based on the aforementioned selective disclosure signature schemes.
- Selective disclosure protocols and systems (clause 6): This clause describes the complete protocols and /or systems that have been developed and standardized based on the aforementioned selective disclosure signature schemes and credential formats.

Since the ARF [i.34] specifies the PID to be issued to an EUDI Wallet as ISO mDL [i.87] (with ISO mDL MSO for selective disclosure) or W3C Verifiable Credentials (with SD-JWT for selective disclosure), these formats and protocols are analysed in more detail in clause 7.

2 References

2.1 Normative references

Normative references are not applicable in the present document.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] Alpar: "U-Prove Cryptography".

[i.2] Altmann: "Inequality tests in salted attribute digest based attestations".

[i.3] Ames-Hazay-Ishai-Venkitasubramaniam: "Ligero: Lightweight Sublinear Arguments Without a Trusted Setup".

[i.4] Asghar: "A Survey on Blind Digital Signatures".

[i.5] Bellare-Goldreich-Goldwasser: "Incremental Cryptography: The Case of Hashing and Signing".

[i.6] Ben-Sasson-Bentov-Horesh-Riabzev: "Scalable, transparent, and post-quantum secure computational integrity (zk-STARK)".

[i.7] Ben-Sasson-Bentov-Horesh-Riabzev: "Scalable zero-knowledge with No Trusted Setup".

[i.8] Ben-Sasson-Chiesa-Genkin-Tromer-Virzs: "SNARKs for C: Verifying Program Executions Succinctly and in zero-knowledge".

[i.9] Ben-Sasson-Chiesa-Riabzev-Spooner: "Aurora: Transparent Succinct Arguments for R1CS".

[i.10] Ben-Sasson-Tromer: "Succinct Non-Interactive zero-knowledge for a von Neumann Architecture".

[i.11] Benjumea-Lopez-Montenegro-Troya: "A First Approach to Provide Anonymity in Attribute Certificates".

[i.12] Boneh-Boyen-Shacham: "Short Group Signatures".

[i.13] Bowe: "BLS12-381: New zk-SNARK Elliptic Curve Construction.

[i.14] Bowe-Grigg-Hopwood: "Recursive Proof Composition without a Trusted Setup".

[i.15] Brands: "A Technical Overview of Digital Credentials".

[i.16] Brands: "Rethinking Public Key Infrastructures and Digital Certificates; Building in Privacy".

[i.17] Brassard-Chaum-Crépeau: "Minimum disclosure proofs of knowledge".

[i.18] BSI TR-03110: "Advanced Security Mechanisms for Machine Readable Travel Documents and eIDAS token".

[i.19] Bull-Stanski-Squire: "Content extraction signatures using XML digital signatures and custom transforms on-demand".

[i.20] Bünz-Bootle-Boneh: "Bulletproofs: Short Proofs for Confidential Transactions and More".

[i.21] Bünz-Fisch-Szepieniec: "Transparent SNARKs from DARK Compilers".

[i.22] Camenisch-Drivers-Lehmann-Neven-Towa: "Short Threshold Dynamic Group Signatures".

[i.23] Camenisch-Lysyanskaya: "A Signature Scheme with Efficient Protocols".

[i.24] Camenisch-Lysyanskaya: "An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation".

[i.25] Camenisch-Mödersheim-Sommer: "A Formal Model of Identity Mixer".

[i.26] Chadwick: "The Use of FIDO2 and Verifiable Credentials".

[i.27] TC/224 WG17: "EN 419 211: Protection profiles for secure signature creation device" (produced by CEN).

[i.28] TC/224 WG20: "New work item: PID onboarding technical standard" (produced by CEN).

[i.29] Chalkias-Cohen-Lewi-Moezinia-Romailler: "HashWires: Hyperefficient Credential-Based Range Proofs".

[i.30] Chaum: "Blind signatures for untraceable payments".

[i.31] Chiesa-Bitansky-Canetti: "From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again".

[i.32] Chiesa-Hu-Maller-Mishra: "Marlin: Preprocessing zkSNARKs with Universal and Updatable SRS".

[i.33] Common Union Toolbox for a Coordinated Approach Towards a European Digital Identity Framework: "The European Digital Identity Wallet Architecture and Reference Framework, Outline".

[i.34] Common Union Toolbox for a Coordinated Approach Towards a European Digital Identity Framework: "The European Digital Identity Wallet Architecture and Reference Framework, Version 1.0.0".

NOTE: The European Digital Identity Wallet Architecture and Reference Framework is commonly called the ARF.

[i.35] Costello-Fournet-Howell et al.: "Geppetto: Versatile Verifiable Computation".

[i.36] Crites: "Delegatable Anonymous Credentials from Mercurial Signatures".

[i.37] Crites-Lysyanskaya: "Mercurial Signatures for Variable-Length Messages".

[i.38] CRYSTALS: "Dilithium digital signature scheme".

[i.39] DIF: "Wallet Security Working Group".

[i.40] Eberhardt-Tai: "ZoKrates - Scalable Privacy-Preserving Off-Chain Computations".

[i.41] EBA (European Banking Association): "Register of payment and electronic money institutions under PSD2".

[i.42] ETSI EN 319 162-1: "Electronic Signatures and Infrastructures (ESI); Associated Signature Containers (ASiC); Part 1: Building blocks and ASiC baseline containers".

[i.43] ETSI EN 319 401: "Electronic Signatures and Infrastructures (ESI); General Policy Requirements for Trust Service Providers".

[i.44] ETSI EN 319 411-1: "Electronic Signatures and Infrastructures (ESI); Policy and security requirements for Trust Service Providers issuing certificates; Part 1: General requirements".

[i.45] ETSI TR 119 001: "Electronic Signatures and Infrastructures (ESI); The framework for standardization of signatures; Definitions and abbreviations".

[i.46] ETSI TS 119 495: "Electronic Signatures and Infrastructures (ESI); Sector Specific Requirements; Certificate Profiles and TSP Policy Requirements for Open Banking".

[i.47] ETSI TS 119 612: "Electronic Signatures and Infrastructures (ESI); Trusted Lists".

[i.48] ETSI TS 119 462: "Electronic Signatures and Infrastructures (ESI); Wallet interfaces for trust services and signing".

[i.49] ETSI TS 119 471: "Electronic Signatures and Infrastructures (ESI); Policy and Security requirements for Electronic Attestation of Attribute Services".

[i.50] ETSI TS 119 472: "Electronic Signatures and Infrastructures (ESI); Profiles for Electronic Attestations of Attributes".

[i.51] European Banking Authority: "Regulatory Technical Standards on strong customer authentication and secure communication under PSD2".

[i.52] Commission Implementing Decision (EU) 2015/1505 of 8 September 2015 laying down technical specifications and formats relating to trusted lists pursuant to Article 22(5) of Regulation (EU) No 910/2014 of the European Parliament and of the Council on electronic identification and trust services for electronic transactions in the internal market.

[i.53] Directive (EU) 2015/2366 of the European Parliament and of the Council of 25 November 2015 on payment services in the internal market, amending Directives 2002/65/EC, 2009/110/EC and 2013/36/EU and Regulation (EU) No 1093/2010, and repealing Directive 2007/64/EC.

NOTE: The Directive (EU) 2015/2366 is commonly called PSD2.

[i.54] Proposal for a regulation of the European Parliament and of the Council amending Regulation (EU) No 910/2014 as regards establishing a framework for a European Digital Identity (June 2021).

NOTE 1: The Proposal for a regulation of the European Parliament and of the Council amending Regulation (EU) No 910/2014 is commonly called eIDAS2.

NOTE 2: The European Council issued an amended edition of eIDAS2 in December 2022 and the European Parliament issued another amended edition of eIDAS2 in February 2023. Unless stated otherwise, the European Commission's edition of eIDAS2 issued in June 2021 is by default the referenced version of the eIDAS2 regulation.

[i.55] Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC.

NOTE: The Directive (EU) 910/2014 is commonly called eIDAS.

[i.56] Eurosmart PP-0117: "Protection Profile for Secure Sub-System in System-on-Chip (3S in SoC)".

[i.57] FALCON: "Fast-Fourier Lattice-based Compacts Signatures over NTRU".

[i.58] Federal Public Key Infrastructure Policy Authority: "X.509 Certificate Policy For The U.S. Federal PKI Common Policy Framework".

[i.59] Gabison-Williamson-Ciobotaru: "PLONK: Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge".

[i.60] Garcia-Rodriguez-Moreno-Bernabe-Skarmeta: "Implementation and evaluation of a privacy-preserving distributed ABC scheme based on multi-signatures".

[i.61] Global Platform: "TEE Protection Profile".

[i.62] Goldwasser-Micali-Rackoff: "The knowledge complexity of interactive proof systems".

[i.63] Groth: "Short pairing-based non-interactive zero-knowledge arguments".

[i.64] Hyperledger Foundation: "AnonCreds Specification v1.0".

[i.65] Hyperledger Foundation: "Hyperledger Aries".

[i.66] Hyperledger Foundation: "Hyperledger Fabric".

[i.67] Hyperledger Foundation: "Hyperledger Indy".

[i.68] Hyperledger Foundation: "Hyperledger Ursa SDK".

[i.69] IBM® Research: "Identity Mixer (IDEMIX)".

[i.70] ICT Trust and Security Research: "Attribute based Credentials for Trust (ABC4Trust)".

[i.71] IETF CFRG: "The BBS Signature Scheme".

[i.72] IETF CFRG: "Pairing-Friendly Curves".

[i.73] IETF IESG: "JOSE and COSE Encoding for Post-Quantum Signatures".

[i.74] IETF JOSE: "JSON Web Proof (JWP)".

[i.75] IETF OAUTH: "SD-JWT-based Verifiable Credentials with JSON payloads (SD-JWT VC)".

[i.76] IETF OAUTH: "Selective Disclosure for JWTs (SD-JWT)".

[i.77] IETF RFC 5652: "Cryptographic Message Syntax (CMS)".

[i.78] IETF RFC 5755: "An Internet Attribute Certificate Profile for Authorization".

[i.79] IETF RFC 7049: "Concise Binary Object Representation (CBOR)".

[i.80] IETF RFC 7515: "JSON Web Signature (JWS)".

[i.81] IETF RFC 7516: "JSON Web Encryption (JWE)".

[i.82] IETF RFC 7519: "JSON Web Token (JWT)".

[i.83] IETF RFC 8152: "CBOR Object Signing and Encryption (COSE)".

[i.84] IETF RFC 8259: "JavaScript Object Notation (JSON) Data Interchange Format".

[i.85] IETF RFC 8610: "Concise Data Definition Language (CDDL): A Notational Convention to Express Concise Binary Object Representation (CBOR) and JSON Data Structures".

[i.86] IETF RFC 9162: "Certificate Transparency Version 2.0".

[i.87] ISO/IEC 18013-5: "Personal identification - ISO-compliant driving licence - Part 5: Mobile driving licence (mDL) application".

[i.88] ISO/IEC 18013-7: "Personal identification - ISO-compliant driving licence - Part 7: Mobile driving licence (mDL) add-on functions".

[i.89] ISO/IEC 23220-4: "Cards and security devices for personal identification - Building blocks for identity management via mobile devices - Part 4: Protocols and services for operational phase".

[i.90] ISO/IEC 23220-6: "Cards and security devices for personal identification - Building blocks for identity management via mobile devices - Part 6: Mechanism for use of certification on trustworthiness of secure areas".

[i.91] ISO/IEC 27001: "Information security, cybersecurity and privacy protection -- Information security management systems -- Requirements".

[i.92] ISO/IEC 27002: "Information security, cybersecurity and privacy protection -- Information security controls".

[i.93] Kampanakis-Panburana-Daw-Van Geest: "The Viability of Post-Quantum X.509 Certificates".

[i.94] Kosba-Papadopoulos-Papamanthou-Song: "MIRAGE: Succinct Arguments for Randomized Algorithms with Applications to Universal zk-SNARKs".

[i.95] Kosba-Papmanthou-Shi: "xJsnark: A Framework for Efficient Verifiable Computation".

[i.96] Lapon-Kohlweiss-Decker-Naessens: "Analysis of Revocation Strategies for Anonymous Idemix Credentials".

[i.97] Maller-Bowe-Kohlweiss-Meiklejohn: "Sonic: Zero-Knowledge SNARKs from Linear-Size Universal and Updatable Structured Reference Strings".

[i.98] Menezes: "An Introduction to Pairing-Based Cryptography".

[i.99] Microsoft® Research: "Spartan: High-speed zkSNARKs without trusted setup".

[i.100] Microsoft® Research: "U-Prove".

[i.101] Mouris-Tsoutsos: "Zilch: A Framework for Deploying Transparent Zero-Knowledge Proofs".

[i.102] Nitulescu: "zk-SNARKs: A Gentle Introduction".

[i.103] NIST: "Digital Identities - Mobile Driver's License (mDL)".

[i.104] NIST: "Post-Quantum Cryptography (PQC)".

[i.105] OpenID Foundation: "OpenID Connect Core 1.0".

[i.106] OpenID Foundation: "OpenID for Verifiable Presentations".

[i.107] OpenID Foundation: "Self-Issued OpenID Provider v2".

[i.108] Parno-Howell-Gentry et al: "Pinocchio: Nearly Practical Verifiable Computation".

[i.109] Petkus: "Why and How zk-SNARK Works: Definitive Explanation".

[i.110] Pointcheval-Sanders: "Short Randomizable Signatures".

[i.111] PrimeLife: "Identity Mixer".

[i.112] Radboud University Nijmegen: "IRMA project".

[i.113] Rivest-Shamir: "PayWord and MicroMint: Two simple micropayment schemes".

[i.114] Setty: "Spartan: Efficient and general-purpose zkSNARKs without trusted setup".

[i.115] SOG-IS Crypto Working Group: "SOG-IS Crypto Evaluation Scheme Agreed Cryptographic Mechanisms".

[i.116] SPHINCS+: "Stateless hash-based signature scheme".

[i.117] Steinfeld-Bull-Zheng: "Content Extraction Signatures".

[i.118] U.S. Department of Homeland Security: "Cryptography Review of W3C Verifiable Credentials Data Model (VCDM) and Decentralized Identifiers (DIDs) Standards and Cryptography Implementation Recommendations".

[i.119] Wahby-Setty-Ren-Blumberg-Walfish: "Efficient RAM and Control Flow in Verifiable Outsourced Computation".

[i.120] Wahby-Tzialla-Shelat: "Doubly-Efficient zkSNARKs Without Trusted Setup".

[i.121] W3C®: "Decentralized Identifiers (DIDs) v1.0".

[i.122] W3C®: "Json Web Proofs for Binary Merkle Trees".

[i.123] W3C®: "Merkle Disclosure Proof 2021".

[i.124] W3C®: "Remove securing JSON, VC-JWT issue #88".

[i.125] W3C®: "Securing Verifiable Credentials using JSON Web Tokens".

[i.126] W3C®: "Universal Wallet 2020".

[i.127] W3C®: "Verifiable Credentials Data Integrity 1.0".

[i.128] W3C®: "Verifiable Credentials Data Model v1.1".

[i.129] W3C®: "Web Authentication: An API for accessing Public Key Credentials Level 2".

[i.130] W3C® CCG: "BBS Cryptosuite v2023".

[i.131] ZKProof: "HashWires: Range Proofs from Hash Functions".

[i.132] Zhang-Xie-Zhang-Song: "Transparent Polynomial Delegation and Its Applications to zero-knowledge Proof".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI TR 119 001 [i.45], ETSI EN 319 401 [i.43] and the following apply:

atomic (Q)EAA: (Qualified) Electronic Attestation of Attribute with a single attribute claim

attribute: feature, characteristic or quality of a natural or legal person or of an entity, in electronic form (as defined in the ARF [i.34])

authentic source: repository or system, held under the responsibility of a public sector body or private entity, that contains attributes about a natural or legal person and is considered to be the primary source of that information or recognized as authentic in national law (as defined in the ARF [i.34])

blind signature: type of digital signature in which the content of a message is disguised (blinded) before it is signed

EXAMPLE: The concept of blind signatures can be exemplified by a voting system in the physical world. The voter encloses an anonymous ballot in a carbon envelope with the voter's name written on the outside. An official verifies the voter's identity and signs the envelope, such that the ballot inside the carbon envelope gets signed with the official's signature. The voter moves the signed ballot to a new unmarked envelope. Hence, the signing official does not see the content of the vote, but a third party can later verify its signature and know that the vote is valid.

NOTE 1: Blinded signatures cater for unlinkability, since the verifier cannot link the signed messages back to the user.

NOTE 2: The U-Prove scheme (clause 6.7) utilizes blinded signatures when issuing the credentials.

Electronic Attestation of Attributes (EAAs): attestation in electronic form that allows the authentication of attributes (as defined in the ARF [i.34])

EUDI Wallet Instance: instance of an EUDI Wallet Solution belonging to and which is controlled by a user (as defined in the ARF [i.34])

EUDI Wallet Provider: organization, public or private, responsible for the operation of a eIDAS-compliant EUDI Wallet Solution that can be instantiated, e.g. through installation and initialization (as defined in the ARF [i.34])

EUDI Wallet Solution: EUDI Wallet Solution is the entire product and service owned by an EUDI Wallet Provider, offered to all users of that solution. An EUDI Wallet solution can be certified as being EUDI-compliant by a CAB (as defined in the ARF [i.34])

ISO mDL: ISO mobile driving license (mDL) according to ISO/IEC 18013-5 [i.87] and ISO/IEC 18013-7 [i.88]

Issuing Authority Certification Authority (IACA): certification authority in the context of ISO mDL that issues certificates for the creation of ISO mDL MSOs and auxiliary certificates for revocation services or securing online services (such as TLS servers)

issuer: issuing authority that is accredited or supervised for issuing certificates, attested attributes, ISO mDL, or credentials

NOTE 1: In the context of eIDAS2, the issuer can be a Person Identification Data Provider issuing PIDs or a (Qualified) Trust Service Provider issuing (Q)EAAs (as defined in the ARF [i.34]).

NOTE 2: In the context of ISO mDL, the issuer is an IACA that issues certificates for the creation and operation of ISO mDL MSOs.

MSO: ISO mobile driving license Mobile Security Object (MSO), with salted hash values of the user's elements in the ISO mDL data

Person Identification Data (PID): set of data enabling the identity of a natural or legal person, or a natural person representing a legal person to be established (as defined in the ARF [i.34])

Person Identification Data Provider (PIDP): Member State or legal entity providing Person Identification Data to users (as defined in the ARF [i.34])

predicate proof: verifiable Boolean assertion (true or false) about the value of another attribute claim in the attestation without disclosing the claim value itself

EXAMPLE 1: Predicate proofs are often in the form of greater than, less than, equal to, set member, etc.

EXAMPLE 2: A user can prove to a verifier that he/she is an EU citizen, without revealing in which Member State.

NOTE: Predicate proofs are often employed in Zero-Knowledge Proof systems aimed at limiting information disclosure.

Qualified Electronic Attestations of Attributes (QEAAs): Electronic Attestation of Attributes, which is issued by a Qualified Trust Service Provider and meets the requirements laid down in eIDAS Regulation amendment proposal Annex V [i.54]

NOTE: A (Qualified) Electronic Attestation of Attribute is abbreviated as (Q)EAA, and is a collaborative term that is used when either a QEAA or an EAA could be applicable for the context.

Quantum-Safe Cryptography (QSC): cryptographic algorithms (typically public-key algorithms) that are expected to be secure against a cryptanalytic attack by a quantum computer

NOTE 1: NIST conducts a research program [i.104] to identify candidates for QSC algorithms that can be standardized. The signature scheme finalists (June 2023) are FALCON [i.57], CRYSTALS Dilithium [i.38] and SPHINCS+ [i.116].

NOTE 2: The term post-quantum cryptography is used in other literature, and is equivalent to the term quantum-safe cryptography that is used throughout the present document.

range proof: method by which the user (prover) can prove to the relying party (verifier) that a number is in a given range (lower and upper bound) without disclosing the actual number

EXAMPLE: A 21 year old user can prove to a verifier that he/she is older than 18 years, without revealing their actual age.

NOTE: Range proofs are subsets of predicate proofs; a range proof is typically generated by using two inequality tests, one for each boundary.

SD-JWT: W3C Verifiable Credential (VC) used in conjunction with a SD-JWT [i.76] with a list of salted hash values of the user's claims in the W3C VC

selective disclosure: capability of the EUDI Wallet that enables the user to present a subset of attributes provided by the PID and/or (Q)EAAs (as defined in the ARF [i.34])

EXAMPLE: Assume that a user's EUDI Wallet includes a (Q)EAA with the attributes first name, last name, birth date, and address. The user can for example selectively disclose only its first name.

NOTE: ISO mDL MSO (clause 7.2) and IETF SD-JWT (clause 7.3) can present selectively disclosed attributes based on the design of salted hashes of the attributes.

unlinkability: lack of information required to connect the user's selectively disclosed attributes beyond what is disclosed

EXAMPLE 1: Assume that a user's EUDI Wallet includes a (Q)EAA with the attributes first name and last name. The user can disclose its first name to one relying party, and its last name to another relying party. The relying parties cannot exchange any information that allows them to link the user's first name disclosure to the last name disclosure.

EXAMPLE 2: The same principle applies if the user discloses its first name to a relying party and later discloses its last name to the same relying party and the single relying party cannot link the user's first name disclosure to its last name disclosure.

EXAMPLE 3: The same principle applies if the issuer colludes with the verifier without being able to link the user's first name disclosure to its last name disclosure.

user: natural or legal person using an EUDI Wallet (as defined in the ARF [i.34])

NOTE 1: In the context of selective disclosure, the user is also the prover of the attributes it presents from its EUDI Wallet.

NOTE 2: The user is sometimes also denoted as holder in other specifications

verified issuer certificate authority list (VICAL) provider: ISO mDL provider that can compile, operate and provide trust anchors (such as IACA trust anchors) in the form of a service to mDL participants

W3C VCDM: W3C Verifiable Credential (VC) Data Model. The W3C VC Data Model v1.1 exists as a recommendation [i.128]

Zero-Knowledge Proof (ZKP): method by which the user (prover) can prove to the relying party (verifier) that a given statement is true while the user does not provide any additional information apart from the fact that the statement is true

NOTE 1: A Zero-Knowledge Proof protocol should meet the following three criteria: Completeness (if the statement is true then a user can convince a verifier), soundness (a fraudulent user can not convince a verifier of a false statement), and zero-knowledge (the interaction only reveals if a statement is true and nothing else).

NOTE 2: A Zero-Knowledge Proof system provides predicate proofs, selective disclosure and unlinkability per definition.

EXAMPLE: Bulletproofs (clause 4.5.1), zk-SNARK (clause 4.5.3) and zk-STARK (clause 4.5.4) are all examples of Zero-Knowledge Proof protocols.

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the abbreviations given in ETSI TR 119 001 [i.45] and the following apply:

3S	Secure Sub-System
AA	Attribute Authority
ABC	Attribute Based Credentials
ARF	Architecture Reference Framework
BBS	Boneh-Boyen-Shacham
BLE	Bluetooth Low Energy
BLS	Barreto-Lynn-Scott (pairing-friendly elliptic curves)
BSI	Bundesamt für Sicherheit in der Informations technik
CBOR	Concise Binary Object Representation
CCG	Credentials Community Group
CDDL	Concise Data Definition Language
CES	Content Extraction Signatures
CFRG	Crypto Forum Research Group
CIR	Commission Implementing Regulation
CL	Camenisch-Lysyanskaya
CLRSA	Camenisch-Lysyanskaya signatures based on RSA
CMS	Cryptographic Message Syntax
COSE	CBOR Object Signing and Encryption
CRYSTALS	Cryptographic Suite for Algebraic Lattices
CS	Computationally Sound

DIF	Digital Identity Foundation
DLP	Discrete Logarithm Problem
DLREP	Discrete Logarithm Representation
dp-ABC	distributed privacy-preserving Attribute Based Credentials
EAA	Electronic Attestation of Attributes
EBA	European Banking Association
EUDI	European Union Digital Identity
EUDIW	European Union Digital Identity Wallet
FALCON	Fast-Fourier Lattice-based Compacts Signatures over NTRU
FIDO	Fast Identity Online
FPKIPA	Federal Public Key Infrastructure Policy Authority
IACA	Issuing Authority Certification Authority
ICAO	International Civil Aviation Organization
IDEMIX	Identity Mixer
IEC	International Electrotechnical Commission
IOP	Interactive Oracle Proof
JAdES	JSON Advanced Electronic Signatures
JOSE	JSON Object Signing and Encryption
JSON	JavaScript Object Notation
JSON-LD	JSON for Linking Data
JWS	JSON Web Signature
JWT	JSON Web Token
mDL	mobile Driving License
MSO	Mobile Security Object
NCCoE	National Cybersecurity Center of Excellence
NTRU	Number Theory Research Unit
OID4VP	OpenID for Verifiable Presentations
OIDC	OpenID Connect
p-ABC	privacy-preserving Attribute Based Credentials
PCP	Probabilistically Checkable Proofs
PID	Person Identification Data
PIDP	Person Identification Data Provider
PII	Personal Identifiable Information
PIOP	Polynomial Interactive Oracle Proof
PKIX	Public-Key Infrastructure (X.509)
PSD2	Payment Services Directive v2
PS-MS	Pointcheval-Sanders Multi-Signatures
QAP	Quadratic Arithmetic Program
QEAA	Qualified Electronic Attestation of Attributes
QSC	Quantum-Safe Cryptography
QTSP	Qualified Trust Service Provider
QWAC	Qualified Website Authentication Certificate
RDF	Resource Description Framework
ROM	Random Oracle Model
RSAREP	RSA Representation
RTS	Regulatory Technical Standard
SD	Selective Disclosure
SD-JWT	Selective Disclosure JSON Web Token
SIOP2	Self-Issued OpenID Provider v2
SoC	System on Chip
SOG-IS	Senior Officials Group Information Systems Security
SSP	Square Span Program
UI	User Interface
VC	Verifiable Credential
VCDM	Verifiable Credential Data Model
VDR	Verifiable Data Registry
VICAL	Verified Issuer Certificate Authority List
VP	Verifiable Presentation
W3C	World Wide Web Consortium
WG	Working Group
XAdES	XML Advanced Electronic Signatures
YAML	Yet Another Multicolumn Layout

ZKP	Zero-Knowledge Proof
zk-SNARK	Zero-Knowledge Succinct Non-Interactive Argument of Knowledge
zk-STARK	Zero-Knowledge Scalable Transparent Argument of Knowledge

4 Selective disclosure signature schemes

4.1 General

The present clause provides an analysis of a set of selective disclosure signature schemes.

The topics for the analysis of each selective disclosure signature scheme are:

- Underlying cryptographic algorithms for selective disclosure, unlinkability and optionally ZKP.
- Maturity of the selective disclosure signature scheme's specification and deployment.
- Cryptographic aspects, more specifically if the cryptographic algorithms used for the selective disclosure signature schemes are approved by SOG-IS and allows for QSC algorithms for future use.

There exist four main categories to enable selective disclosure:

- The first category is using atomic (Q)EAAs, which is described in clause 4.2.
- The second category is using a selective disclosure capable multi-message signature scheme, which typically relies on commitments. This category is explained in clause 4.3.
- The third category is signing a collection of salted attribute digests; this category is described in clause 4.4.
- There is also a fourth category of methods that can ensure the privacy of any computable proof (e.g. Bulletproofs, zk-SNARKS, zk-STARKS etc.). This category is elaborated in clause 4.5. These methods could support additional selective disclosure mechanisms than the three main ones listed above.

NOTE: An argument can be made for a selective disclosure mechanism that relies on trusted components for storage and computation. It is possible to store unsigned attribute claims on trusted storage and transport only the requested claims over a secure messaging channel. It is also possible in these setups to associate each storage partition with a unique key and only store a single (Q)EAA per partition in order to ensure the proper pairing of attributes. A solution based on these principles is detailed in BSI TR-03110 [i.18]. The solutions described in the present document, however, include only signature based selective disclosure schemes.

Each of the four main ways are described in the clauses below.

4.2 Atomic (Q)EAAs schemes

An atomic electronic attribute attestation is a (Q)EAA with a single attribute claim, which can be issued by a (Q)TSP upon request or as part of a batch to an EUDI Wallet. The atomic (Q)EAAs can be selected by the user and be included in a verifiable presentation that is presented to a verifier.

An example of a solution based on atomic (Q)EAAs is illustrated in Figure 2. In this scenario, the user needs a parking ticket to enter a car parking. For that purpose, the user enrolls for atomic (Q)EAAs from a transport authority (with the car registration number), from a civil registry (with the address), and from a payment service provider (with the paid amount). The user's EUDI Wallet can then combine these atomic (Q)EAAs into a verifiable presentation, which is the parking ticket that is presented to the car parking clerk.

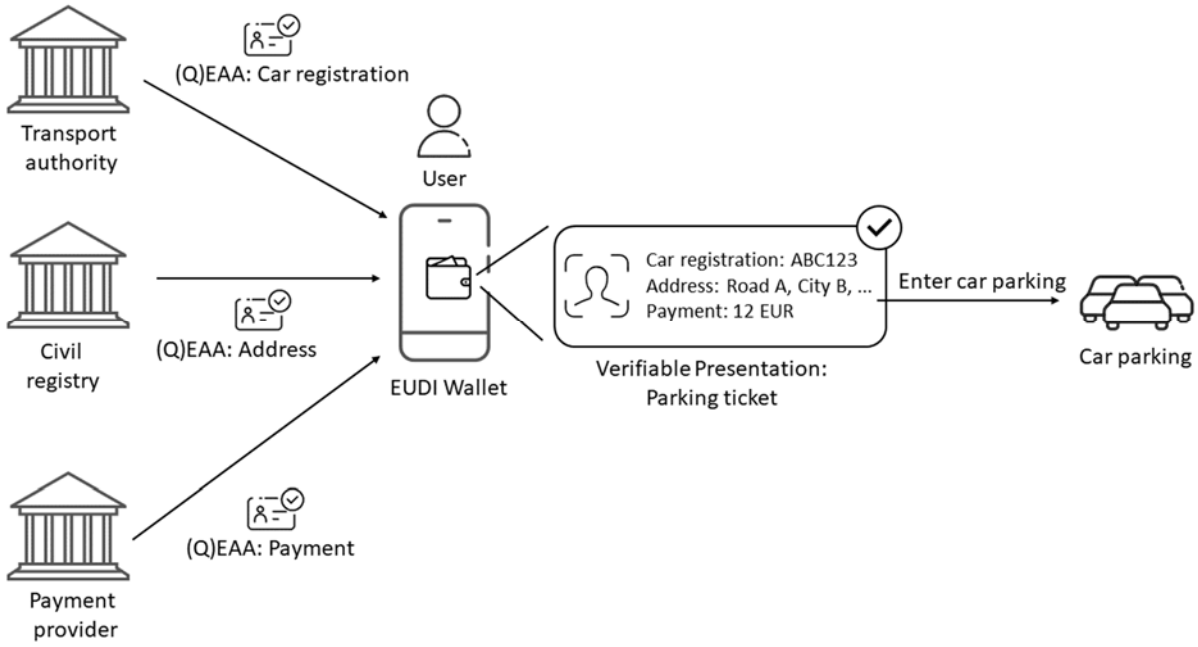

Figure 2: Example of atomic attribute credentials

The underlying cryptographic algorithms depend on the (Q)TSPs' signing algorithms of the (Q)EAAs and the proof key when signing the verifiable presentation. Hence, it is possible to select signature algorithms that are approved by SOG-IS and/or allow for QSC. (More information on the specific (Q)EAA formats X.509 attribute certificates and W3C Verifiable Credentials is available in clauses 5.2.1 and 5.2.2).

Unlinkability can be achieved by enrolling for atomic (Q)EAAs on demand, which results in an unused set of (Q)EAAs with new signatures that cannot be correlated with any previous signatures.

NOTE 1: If the atomic (Q)EAAs are issued batchwise to an EUDI Wallet, it is recommended to keep track of the atomic (Q)EAAs that have been used for presentations, and replace them with new atomic (Q)EAAs.

NOTE 2: Atomic attribute credentials cannot alone guarantee that the claims are paired properly in a presentation. For instance, if the user has a credential from the civil registry with an address, and one for their company they are the legal representative of, there is nothing preventing the user from creating a presentation that improperly pairs the company's address with the user's private car registration. Verifiers cannot trust that verifiable presentations containing multiple atomic attribute credentials are properly paired without additional mechanisms preventing improper pairing.

4.3 Multi-message signature schemes

4.3.1 Boneh-Boyen-Shacham (BBS) signature scheme

4.3.1.1 Academic research on the BBS signature scheme

The BBS signature scheme was initially invented in 2004 by the Stanford cryptography research team Dan Boneh, Xavier Boyen, and Hovav Shacham, who also named the BBS signature scheme after their initials. It is based on elliptic curve cryptography bilinear pairings [i.98].

The BBS signature scheme [i.12] is a multi-message signature scheme that can preserve the algebraic structure of the messages and is capable of selective disclosure. As many similar multi-message capable schemes, BBS supports both (blind) signing of multiple messages (while outputting only a single digital signature) and predicate proofs (see clause 5.3.2 on BBS signed W3C VC and the support for predicate proofs). A user who possesses a signature is able to generate multiple proofs that selectively disclose subsets of the originally signed messages, yet preserving the authenticity and integrity of the messages.

A user can generate a ZKP proof of knowledge of a valid BBS signature, which makes BBS signatures suitable in cases that seek to prevent linkability through the issuer's signature.

4.3.1.2 Overview of BBS

The BBS signature scheme is illustrated in Figure 3.

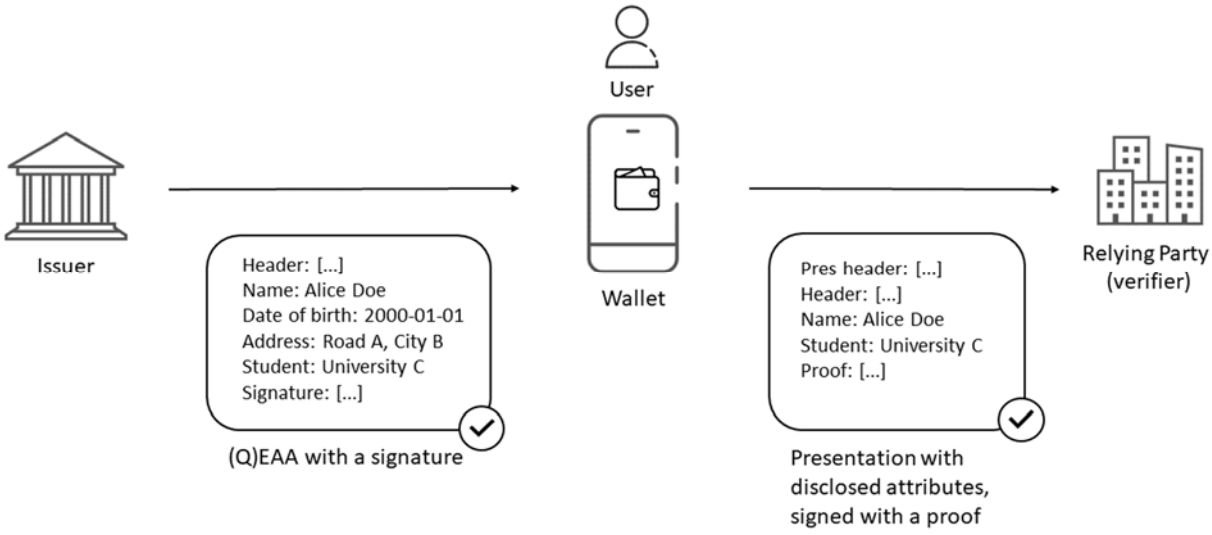

Figure 3: Overview of the BBS signature scheme

The issuer issues a (Q)EAA, with a header and a complete set of attributes, which is signed by the issuer. The (Q)EAA is stored in the user's wallet.

The user selects the attributes to disclose to a relying party, and the wallet generates a presentation with the disclosed attributes. The presentation contains a presentation header, the original header, the selectively disclosed attributes, and a proof. The proof reveals the user's knowledge of the original signature, but does not reveal the actual signature.

4.3.1.3 IETF CFRG BBS specification

The IETF Crypto Forum Research Group (CFRG) has created an internet draft specification of the BBS signature scheme [i.64]. The specification describes the following topics:

- Scheme Definition ([i.64], clause 3) defines the core operations and parameters for the BBS signature scheme.
- Utility Operations ([i.64], clause 4) defines utilities used by the BBS signature scheme.
- Security Considerations ([i.64], clause 5) describes a set of security considerations associated with the signature scheme.
- Ciphersuites ([i.64], clause 6) define the format of a ciphersuite.

More specifically, the IETF BBS draft specifies pairing-friendly ECC curves [i.72] alongside a concrete ciphersuite based on the BLS12-381 curve.

4.3.1.4 Cryptographic analysis of the BBS signature scheme

However, ECC algorithms based on bilinear pairings are vulnerable against quantum computing attacks [i.118]. This is an identified weakness of the BBS signature scheme, which has been described in a cryptographic review [i.118] prepared for the U.S. Department of Homeland Security [i.118]. The report [i.118] claims that BBS signatures are not standardized by NIST, and are unlikely to be standardized, since they rely on ECC with BLS12-381 curves that are not considered quantum-safe. The cryptographic review [i.118] gives the following recommendations for the IETF CFRF BBS draft specification to move closer to government compliance: use the SHAKE256 hash function from SHA-3 and an approved random number generator in the BBS signature implementation.

NOTE: SOG-IS has not approved the BLS12-381 [i.13] curves either, meaning that the BBS signature scheme cannot be considered to be compliant as a cryptographic algorithm for use with the EUDI Wallet yet.

4.3.2 Camenisch-Lysyanskaya (CL) signatures

4.3.2.1 Introduction to CL-signatures

In their paper "A signature scheme with efficient protocols" [i.23] (2002), Camenisch and Lysyanskaya introduce the CL-signature. The authors explicitly sought to design signature schemes that would be "suitable as building blocks for other applications".

Of particular relevance to this text is that the CL-signature allows for the implementation of two additional protocols. The first protocol is a secure multiparty computation protocol that allows an issuer to issue a signed attestation to the user, without the issuer learning all the message content or the final signature value. The ability for a signer to obliviously sign a user provided commitment to a message is enables, among other things, the user to convenience a verifier that two attestations were issued to the same identity subject simply by providing an equality proof between the two (blinded) commitments in the two attestations. Relatedly, it allows the user to generate a proof of possession of the commitment value in a privacy preserving way. The second protocol enables the user to prove possession of a, potentially hidden and blinded, message-signature pair (in CL-signatures, this proof is done in a ZKP manner). This ability for the user to present different looking presentations based on the same underlying issuer signed attestation is an important property when seeking to achieve privacy across distinct authentications.

Together, the two protocols above are introduced to achieve what Camenisch and Lysyanskya describe as an anonymous credential system. Such a system has two important requirements:

1) The user is required to demonstrate to a verifier that they possess the right attributes for a specific service, without the verifier being able to infer anything other than the fact that the user has the right attributes.

2) The user is required to obtain attribute attestations without revealing their identity to the issuer (in the paper "A signature scheme with efficient protocols" [i.23], the authors consider the user's secret key to be equivalent to the user's identity).

A signature scheme that can meet the above two requirements is one that allows the design of protocols that can prove statements in the form of "I have a valid signature" and where these signatures are over blinded committed values.

4.3.2.2 The CL-signature scheme

CL signatures enable signing of messages without affecting the message's algebraic structure; a property that allows a user to prove statements about messages even if these messages are hidden in some way (e.g. using a commitment).

For key generation, the first CL-scheme relies on a special RSA modulus n=pq, where (p, q) are safe primes, and the quadratic residues mod n (a,b,c). The public key is (n,a,b,c) and the secret key is (p). The message space consists of the integers in range $[0,2^{l\_m})$ for the parameter l_m. The signing algorithm takes as input a message m, selects a random prime number e and a random value s of suitable lengths (the paper "A signature scheme with efficient protocols" [i.23] details how to select the proper parameters) and computes the value v such that $v^e = a^m b^s c \ (mod\ n)$. The signature verification is done using the tuple (e,s,v), where it is the user that completes the value for s based on input from the issuer, and the message m by checking that $v^e = a^m b^s c \ (mod\ n)$ and that e is within the suitable range.

Later versions rely on pairings and are more efficient. Note however that no CL-signature scheme is quantum-safe nor possible to construct using SOG-IS approved inputs.

As aforementioned, the CL-signature scheme preserves the message's algebraic structure. As such, when signing a block of messages, $(m_1, m_2, \ldots, m_L)$ it is not permitted to simply sign the hash over the block of messages $H(m_1, m_2, \ldots, m_L)$ as this would make it impossible to both prove relations among the message components, the oblivious signature demand, and to prove predicates. Instead, the previous signing algorithm is modified to allow for multi-message signing as follows:

$$v^e = a_1^{m_1} a_2^{m_2} \ldots a_L^{m_L} \ \ b^s c \ (mod\ n)$$

Next it will be described how the CL-signature scheme enables selective disclosure.

4.3.2.3 The CL-signature scheme and selective disclosure

In essence, the CL-signature includes a commitment vector of messages $a_1^{m_1} a_2^{m_2} \ldots a_L^{m_L}$. The following characteristics can now be observed:

- All the quadratic residues are public.
- The commitment $a_i^{m_i}\ (mod\ n)$ prevents the verifier from learning m_i as long as solving the discrete log problem in that group is hard.
- The user can present any combination of the commitment and the cleartext message.

The last point is what enables selective disclosure. Basically, the user will present in cleartext all the messages they wish to reveal, and the commitments to the messages they wish to keep secret. For instance, if a user wants to present m_1 but keep m_2 hidden, the user would present $((a_1, m_1), a_2^{m_2})$.

4.3.2.4 The CL-signature scheme, predicates, and knowledge proofs

Since the algebraic structure of the messages is preserved, it is possible to generate various proofs using CL-signatures.

In their original paper, Camenisch and Lysyanskya list the following protocols known to be secure under the strong RSA assumption:

- Proof of knowledge of discrete logarithm representation modulo a composite. Under specific conditions, this can be used to prove knowledge of exponents $(m_1, m_2, \ldots, m_L)$ in the commitments $a_1^{m_1} a_2^{m_2} \ldots a_L^{m_L}$ without revealing the exponents.
- Proof of knowledge of equality of representation modulo two (possibly different) composite moduli. This one is similar to the one above, but can prove knowledge of exponents even if the bases are different and the composite moduli are different.
- Proof that a committed value, $g^{ab} h^{r_3} (mod\ n)$, is the product of two other committed values, ($g^a h^{r_1} (mod\ n)$, $g^b h^{r_2} (mod\ n)$), without revealing any of the values.
- Proof that a committed value, $g^x h^r (mod\ n)$, lies in a given integer interval $a \leq x \leq b$. This builds on other known proofs that a committed value is a square (i.e. a positive number) and greater than or equal to proofs.

The above support the various predicate proofs that attestation systems based on CL-signatures are capable of, set (non-)membership tests, enable the property where the user can provide a proof of a valid signature as opposed to presenting the signature itself, and allows the user to request a signature over blinded messages. By extension, these properties provide unlinkability for the user as issuer and verifiers cannot collude to track use of an attestation.

EXAMPLE: A positive number proof can be easily constructed using other proofs. Lagrange's four-square theorem states that every natural number can be represented as the sum of four non-negative integer squares. Remember that there exists a way for the user to prove that a committed value is a square. A user could then send over the commitments to the square values, together with their corresponding proofs. The verifier can then easily check that another number is a positive number using the four commitments of a square number proof.

Recently, Thomas Groß extended the CL-signature scheme to obtain a signature on a committed graph and demonstrated that there exists a proof system on graph 3-colorability, meaning that there exists a CL proof system for all NP problems.

4.3.3 Mercurial signatures

Mercurial signatures cater for privacy preserving schemes, such as anonymous credentials, delegatable anonymous credentials, and related applications. They allow a signature s0 on a message m0 under a public key pk0 to be transformed into a signature s1 on an equivalent message m1 under an equivalent public key pk1. For example, pk0 and pk1 may be unlinkable public keys of the same user, and m0 and m1 may be unlinkable pseudonyms of a user to whom some capability is delegated. Mercurial signatures were presented by Crites-Lysyanskaya [i.36] in 2019.

Mercurial signatures are based on Decisional Diffie-Hellman (DDH) over equivalent groups, and are therefore not considered as plausible quantum-safe cryptography.

4.3.4 Pointcheval-Sanders Multi-Signatures (PS-MS)

Pointcheval-Sanders Multi-Signatures (PS-MS) [i.110] have certain properties that can be used for distributed privacy-preserving Attribute Based Credentials (dp-ABC). The PS-MS signatures are based on a variant of CL-signatures with pairing-friendly curves such as BLS12-461. There is a formal definition of PS-MS signatures by Camenish et al in the paper "Short Threshold Dynamic Group Signatures" [i.22] (2020), which are secure under bilinear group model and random oracle model.

An dp-ABC scheme based on PS-MS signatures has been designed by García-Rodríguez et al in their paper "Implementation and evaluation of a privacy-preserving distributed ABC scheme based on multi-signatures" [i.60] (2021).

The workflow of a dp-ABC scheme is illustrated in Figure 4.

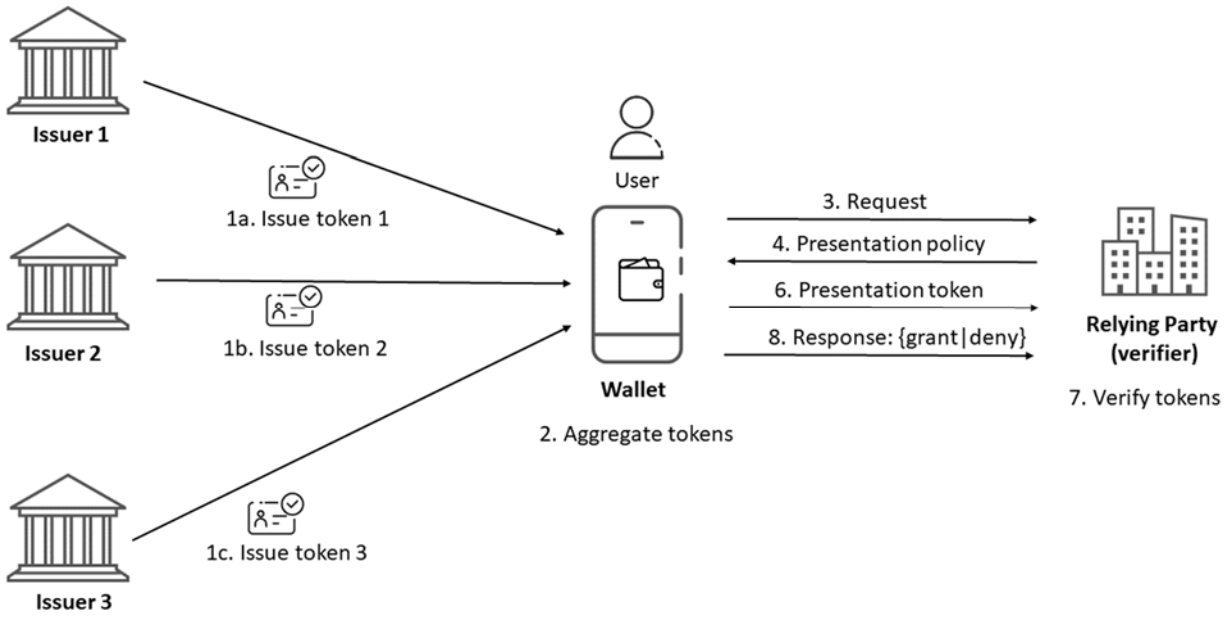

Figure 4: Overview of PS-MS signatures used for dp-ABC flow

More specifically, the PS-MS signatures are used when aggregating the issued tokens in step 2. Selective disclosure and unlinkability is an integral feature of the PS-MS signatures.

NOTE: The identity systems Idemix (clause 6.4) and U-Prove (clause 6.7) are also based on p-ABC schemes, however, they are based on CL-Signatures and the Discrete Logarithm Problem (DLP).

Since the PS-MS signature scheme is based on bilinear-pairings, it is not approved by SOG-IS or considered as being plausible quantum-safe cryptography.

4.4 Hashes of salted attributes

4.4.1 Overview of salted hashes of attributes

Salted hashes of attributes is a concept whereby each attribute in a (Q)EAA is represented by a salted hash value in an indexed list. Hence, the issuer needs to issue the (Q)EAA with the attributes in clear text, along with an indexed list with salts and salted hashes of each attribute.

An overview of salted hashes of attributes is illustrated in Figure 5.

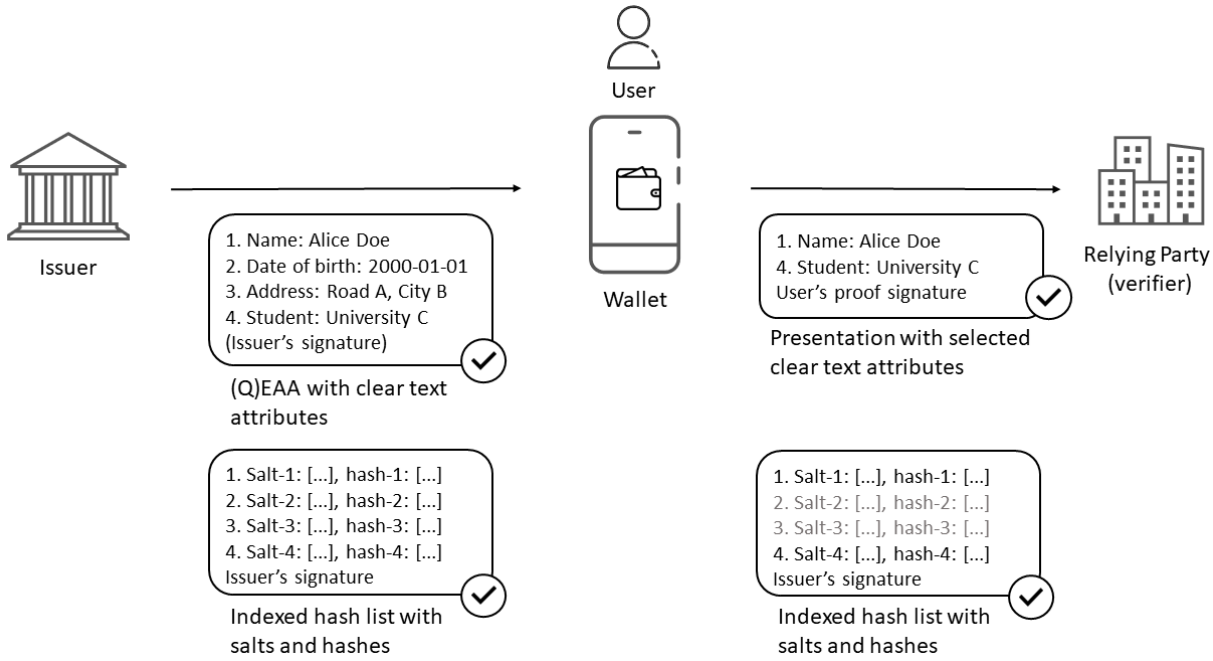

Figure 5: Overview of salted hashes of attributes

In the example above, the issuer issues a (Q)EAA with all attributes in clear text. The issuer also issues an indexed hash list in which each (Q)EAA attribute is represented as a key (index), a random salt, and a hash value over the salt and attribute. The (Q)EAA and indexed hash list are signed by the issuer.

The (Q)EAA and indexed hash list are stored in the user's wallet. The user selects the attributes to disclose to a relying party, and the wallet generates a presentation with the disclosed attributes; the user signs the presentation with its proof key.

The wallet submits the presentation with selected attributes (in clear text) along with the indexed hash list. The relying party parses out the salted hashes from the indexed hash list, and compares them with the salted hashes of the presented attributes.

Solutions based on the concept of salted hashes of attributes have been standardized as IETF SD-JWT and ISO mDL MSO. More information on the specific formats IETF SD-JWT and ISO mDL MSO that use salted hash values for selective disclosure is available in clauses 5.4.1 and 5.4.2.

4.4.2 Issuance phase

The issuance phase of this selective disclosure scheme is in principle based on the following algorithm:

1) Parse out each attribute from a user's (Q)EAA.

2) Concatenate each attribute with a salt, denoted as (salt|attribute).

3) Hash each (salt|attribute), denoted as hash(salt|attribute).

4) Put all the hash(salt|attribute) values and the salts in an indexed hash list, which is signed. The indexed hash list can be expressed as this formula: signed({key-1, salt-1, hash(salt-1|attribute-1)}, … {key-*n*, salt-*n*, hash(salt-*n*|attribute-*n*)}).

5) Store the (Q)EAA in an EUDI Wallet along with the indexed list from step 4.

NOTE 1: The hash algorithm used in step 3 should be listed in the SOG-IS list of approved hash algorithms [i.115], such as SHA-256 or higher.

NOTE 2: The signature algorithm used in step 4 should be listed in the SOG-IS list of approved signature algorithms [i.115], such as ECDSA with BrainpoolP256r1.

NOTE 3: The signature format used in step 4 should allow for QSC algorithms. For example, JOSE and COSE allows for QSC algorithms.

4.4.3 Presentation and verification phase

When presenting selective disclosed attributes in the (Q)EAA along with the indexed list, the relying party can perform the following verification process:

1) The EUDI Wallet parses out the disclosed attribute with key-*x* from the (Q)EAA.

2) The EUDI Wallet submits the disclosed (Q)EAA attribute with key-*x* from step 1 along with the indexed hash list to the relying party. The indexed hash list has the format: signed({key-1, salt-1, hash(salt-1|attribute-1)}, … {key-*n*, salt-*n*, hash(salt-*n*|attribute-*n*)}).

3) The relying party verifies the signature of the indexed hash list from step 2. If the signature check fails, the verification process is stopped, else it continues at step 4.

4) The relying party parses out salt-*x* from the indexed hash list.

5) The relying party parses out hash(salt-*x*|attribute-*x*) from the indexed hash list.

6) The relying party concatenates the disclosed (Q)EAA attribute from step 2 with the corresponding salt-*x* from step 4, and hashes the result.

7) The relying party checks if the result in step 6 is equal to the hash(salt-*x*|attribute-*x*) from step 5. If the values match, the verification process has succeeded.

4.4.4 Unlinkability

In order to prevent correlation based on the salt value, the salts in issuance step 2 should be randomly generated unique values and used only once in a presentation. Consequently, the indexed list in issuance step 4 is also updated. Using unique salts will prevent the comparison of the signatures and salts of previously shared indexed hash lists.

NOTE: Using unique salts, an issuer can always uniquely identify a user from a single disclosed salted attribute. Consequently, salted attribute digests represent a tradeoff between issuers' and verifiers' ability to link together attestation usage. That tradeoff is unproblematic in contexts where issuers are assumed trusted, but represents a great risk in contexts where issuer collusion is possible.

4.4.5 Cryptographic analysis

The (Q)EAA and indexed hash list are separate objects that can be signed with cryptographic algorithms that are approved by SOG-IS [i.115]. In other words, there are no specific requirements on ECC curves for bilinear pairings.

This concept also caters for the (Q)EAA and indexed hash list to be signed in the future with QSC algorithms such as CRYSTALS Dilithium [i.38], FALCON [i.57], or SPHINCS+ [i.116] as discussed in the IETF report "JOSE and COSE Encoding for Post-Quantum Signatures" [i.73].

4.4.6 Predicates based on computational inputs

Salted attribute hashes do not support dynamic calculation of predicates (e.g. to compute a proof for age over 18 given only the birth date and current date). The recommendation is to include boolean claims such as `"age_over_NN": "True"`. However, there is a possibility for the issuer to sign the parameters and the inputs to an inequality test. This would enable the user and the verifier to compare numbers and perform range proofs.

Many techniques in the present document were originally designed with the adversarial assumptions of blockchains in mind. But for an (Q)EAA system, there is normally a) a trusted issuer, and b) a limited need to perform operations between encrypted values (thus eliminating the need for commitment homomorphism and the ability to perform algebraic manipulations). The former is rather self-evident. The latter is because it is normally interesting to prove that an attribute claim satisfies a threshold or inequality and absolutely nothing else. Furthermore, there is a trusted issuer and there is also only the need to hide the exact amount of the values and thus do not care about the ZKP property.

EXAMPLE: The issuer could compute the commitment s = H (seed) and assign this to the user's birth year. The issuer then computes the commitment c = H^k (salt || s), which is k repeated iterations of H. The value for k can be computed e.g. based on the maximum year supported in the calculation. The issuer includes s and c in the signed attestation both as disclosures (the user should never reveal s, only c). The user can now generate an age over 18 proof by constructing a hash chain where the length of the chain equals the k iterations used to arrive at the signed commitment c if and only if the user is above a certain age. Example code is provided in Appendix C. Research on efficient protocols for hash chain based range proofs is underway with one notable example being HashWires [i.131]. And variations of the technique exist that would allow a user to generate a valid age_over_N proof from an age_over_M proof where M > N. The algorithm for HashWires in combination with hashed salted attributes is described in clause 4.5.2.4.

4.5 Proofs for arithmetic circuits

4.5.1 Bulletproofs

In their paper, "Bulletproofs: Short Proofs for Confidential Transactions and More" [i.20], Bünz et al. (2017) introduce a non-interactive ZKP protocol aimed to address the issue of transaction size and verification time in existing privacy preserving protocols. Specifically aiming to improve upon proposals for confidential transactions in cryptocurrencies, bulletproofs support aggregation of range proofs and require no trusted setup.

Bulletproofs allow a prover to convince a verifier that the commitment $C(x, r) = xH + rG$ contains a number x s.t. x lies in the range $[0, 2^{n-1}]$. Since these commitments preserve the algebraic structure of the input messages, it is possible to perform certain operations with the commitments, e.g. to prove that the sum of the input balances equals that to the total output balance. Bulletproofs support range proofs where a party can prove that m commitments lie in a given range as opposed to having to generate a range proof for each commitment. Relatedly, a verifier can verify multiple range proofs quickly (read about the same time as verifying an ECDSA signature).

While the details of how this range proof is achieved is outside the scope of this text, it is here sufficient to mention that it relies on a random linear combination of constraints (to ensure that the input values to the vectors are 0 and 1) and challenges to the prover. The prover then constructs a vectorised inner production relation containing the elements, constraints, and challengers and a blinding vector. The way this relation is constructed leads to a proof size that is very short compared to alternative range proofs.

The primary goal of Bulletproofs is to provide a compact and efficient way of proving the correctness of a transaction, while hiding the specific details of the transaction itself. Bünz et al. (2017) do mention other uses, including support for arithmetic circuits, verifiable shuffles (i.e. to prove that one list of committed values is a shuffle of another list of committed values), and privacy preserving smart contracts in public blockchains. Each of these uses, however, can be done more efficiently in contexts with different contextual characteristics than those of decentralized cryptocurrencies. It is not immediately apparent how bulletproofs are relevant for electronic attestations of attributes/person identification data. Further exploration or analysis may be needed to fully understand how Bulletproofs could be directly applicable to electronic attestations of attributes or person identification data.

4.5.2 HashWires

4.5.2.1 Introduction

In their 2021 paper "HashWires: Hyperefficient Credential-Based Range Proofs", Chalkias et al. [i.29] present a hash based protocol for performing inequality tests (and by extension range proofs) in contexts where a trusted issuer can sign commitments to computational inputs. The computational inputs in HashWires is a commitment c to a hash chain, and the parameter is the hashing algorithm used to create the chain.

HashWires are inherently less flexible than general ZKP inequality tests and range proofs, and do not support homomorphic operations on commitments. However, the commitment and proof conditions, together with the adversarial assumptions in their deployed contexts (e.g. cryptocurrencies), often makes ZKP inequality tests and range proofs unsuitable for resource constrained environments and unnecessarily complex given the presence of a trusted PID/(Q)EAA Provider (as opposed to self signed claims). Put differently, many existing ZKP inequality tests and range proofs were designed to cater for highly adversarial cryptocurrency contexts without any trusted parties or central authorities, and where the user self issues a signed intent to perform a certain transaction. In contrast, HashWires were designed to specifically cater for the needs of the issuer-holder-verifier model. The authors introduce the concept of "Credential-based range proofs" to distinguish these inequality tests and range proofs from their ZKP counterparts.

HashWires is based on the core idea that the trusted third party, i.e. the PID/(Q)EAA Provider, generates and signs the commitment needed for an inequality test. The idea to rely on a trusted third party to sign a commitment can be traced back to Rivest and Shamir's 1996 work on micro-payments. In their paper "PayWord and MicroMint: Two simple micropayment schemes" [i.113], Rivest and Shamir describe how issuer signed hash chains type commitments can be used for payments. A description of their original idea follows.

4.5.2.2 Using a hash chain for inequality tests

A fundamental building block in HashWires is hash chains. Given two collision-resistant hash functions (H, G), a maximum integer value N, and a random value r, the issuer computes the commitment $c = H^k(G(r))$. Here, $H^k(\cdot)$ represents k iterations of the function H s.t. the digest of H^i is the pre-image to H^{i+1}. The issuer signs c and sends (c, r) to the user (optionally also k). The user can now produce a hash chain of the same length as a threshold t by computing the range proof $\pi = H^{k-t}(G(r))$. The user signs a presentation containing (π) and the verifier checks if $c = H^t(\pi)$. If the check passes, the verifier knows that c is the commitment to some value $t \leq x$ but does not learn k.

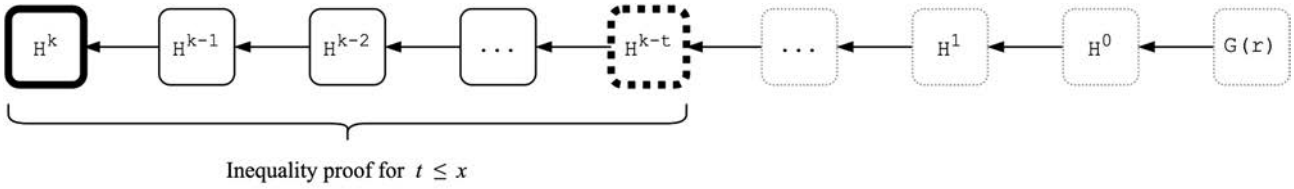

Figure 6: A hash chain based inequality test

In Figure 6, the issuer signs the leftmost bold box representing the commitment $c = H^k(G(r))$. The user presents the dotted bold lined box representing the threshold value $\pi = H^{k-t}(G(r))$. The verifier accepts π as a proof for the inequality $t \leq x$. Note that for an age proof, the value H^0 should represent the user's actual age k at the time of issuance and that H^k represents the minimum age value 0.

NOTE 1: The hash functions (H, G) should be listed in the SOG-IS table of agreed hash functions [i.115].

NOTE 2: The digital signature scheme should be listed in the SOG-IS table of agreed signature schemes [i.115].

NOTE 3: The use of digital signatures that are QSC should be possible.

NOTE 4: The verifier does not learn the value $k, G(r)$ and any $H^m(\cdot)$ where $m > t$.

NOTE 5: A single hash function with two different salts, or a keyed HMAC with two keys, are both alternatives to (H, G).

When considering non-negative integers, one obvious representation is that the H^0 digest represents the maximum value, and each subsequent digest represents a decrement by 1. The problem with that approach is that it does not scale. Take for instance age over or equal to proofs. Here, the user should be able to prove that their age is equal to or above 18 the very day they turn 18, but not before. A hash chain for 18 years in days requires roughly 6 575 digests. This is further exacerbated by the batch issuance requirement for PIDs and (Q)EAAs to prevent verifier collusion (the Provider would need to create a new hash chain for every attestation since the commitment would be correlatable even with a salt). Also, each verifier needs to recompute the threshold length of the chain at every presentation. With ~450 million EU citizens, and potentially multifold more inequality tests for age based services, optimization is required.

4.5.2.3 Using multiple hash chains for inequality tests

The optimization presented in the HashWires paper ensures that the commitment generation, proof and verification, and proof size all scale well even for very large *n*-digit numbers. The core idea is to rely on multiple hash chains. However, instead of representing decrements starting from the maximum number, each digest represents the commitment to the digits x_i of a number $x = x_n \cdot 10^n + x_{n-1} \cdot 10^{n-1} + \ldots + x_1 \cdot 10^1 + x_0$.

For instance, using the commitments to the coefficients in $22 = 2 \cdot 10^1 + 2 \cdot 10^0$ a user could generate a proof for the inequality $x \geq 10$. Note, however, that the user would not be able to use that commitment to prove $x \geq 13$ without revealing a lot more information than necessary (more specifically, the user would need to reveal commitments to 20).

Chalkias et al. here describe the idea of Minimum Dominating Partitions (MDP) to address the above problem. In the HashWires paper, there is a formal definition of MDP, which relies on the idea that a number x dominates another number y if each digit $x_i \geq y_i$. The authors present an algorithm that takes a non-negative integer as input and outputs one or more non-negative integers that represent numbers that dominate other numbers, where the collection of numbers output can dominate any other number in the entire range of the requested inequality.

A simpler explanation is that the MDP is generated using a recursive function that takes as input a number, and outputs the first number that the input cannot dominate. That new output number then becomes the new input number, and the MDP outputs the value it cannot dominate. For instance, using base 10, the number 84 can dominate $\{84,83,82,81,80\}$ but not 79. Subsequently, 79 can dominate all numbers down to 0. So the $MDP(84) = \{84,79\}$. Similarly, $MDP(3413) = \{3413,3409,3399,2999\}$.

Given a set of MDP partitions, the user can use hash chains to dominate any number that up to and including the first element by simply picking the element that can dominate the requested threshold value. For instance, given $MDP(3413) = \{3413,3409,3399,2999\}$ the user can use the $\{2999\}$ element to prove $x \geq 376$. When the user can use more than a single element from the MDP to dominate the threshold number, the user picks the number that reveals the least amount of information.

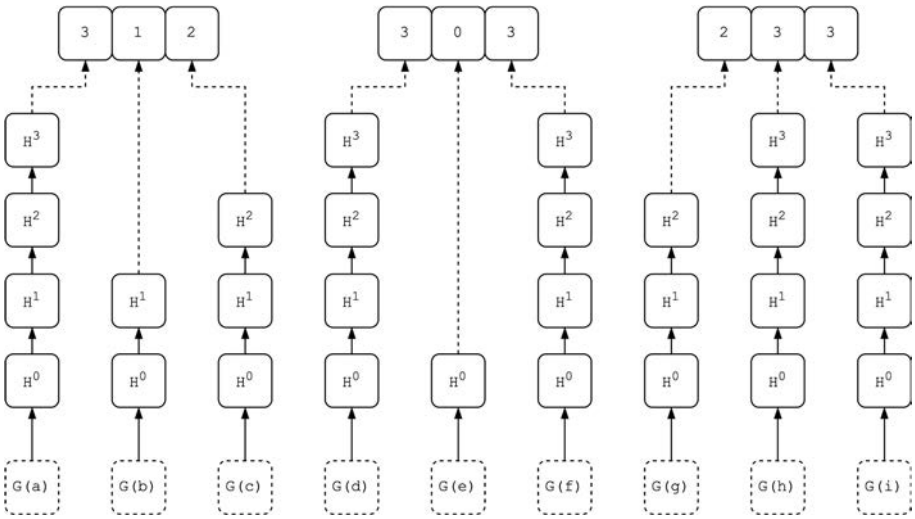

Figure 7: Basic HashWires commitment

Figure 7 illustrates a basic HashWires commitment to the number 312 in base 4 with $MDP_4(312) = \{312,303,233\}$. Each hash chain represents a commitment to a specific digit in each MDP partition.

A further optimization can be made by reusing the same hash chain for multiple different commitments. The idea here is to generate one hash chain per digit in the largest number, with the length of the hash chain being the largest value of any digit in any MDP partition.

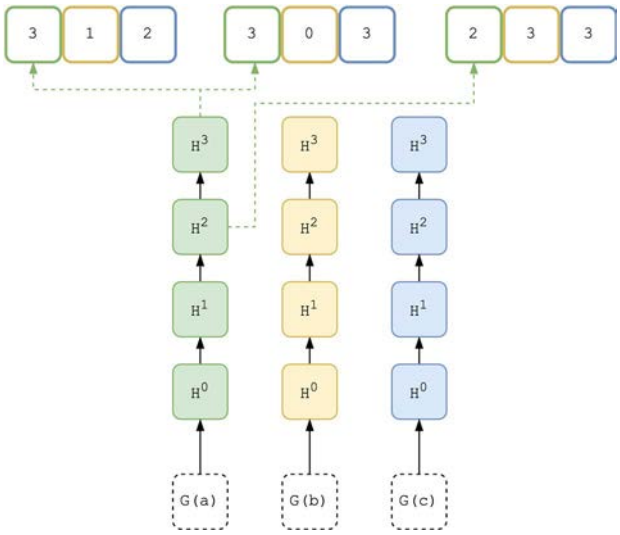

Figure 8: Optimized HashWires commitment

Figure 8 shows an optimized HashWires commitment to the number 312 in base 4 with $MDP_4(312) = \{312,303,233\}$. Each hash chain represents the commitments to the digit values of each partition. Green dotted line illustrates how the values are sourced for the third digit in each MDP partition. Hash chains are colored to correspond to their commitments, i.e. the second digit in each MDP partition would source their commitment from the middle hash chain, and the first digit in each partition would source commitments from the rightmost hash chain.

The optimized HashWires approach is orders of magnitude more efficient than using a single hash chain. Specifically, the $MDP(6575) = \{6575,6569,6499,5999\}$ (18 years in days), requires $3 + 6 + 9 + 9 + 9 = 36$ hash operations (three for the seeds, and then 6 for the fourth digit, and then 9 for each subsequent digit). In fact, using base 10, the maximum possible number of hash chains will never exceed the number of digits multiplied by 10.

One concern with the optimized HashWires approach is that it may leak information about the partitions, and thus reveal the users actual number. To avoid such leaks, the authors of the HashWires paper suggest the use of an accumulator that can hide the actual commitments. While the use of an accumulator addresses the concern, it is also not necessary when the attestation format is capable of selectively disclosing the particular commitment that the user needs to prove the inequality, and when attestations are batch issued and used only once (that is not to say that the issuer cannot select to include the accumulator value as a selectively disclosable value).

4.5.2.4 Protecting optimized HashWires with SD-JWT or MSO

The MDP partitions leak information about the number in several ways. Therefore, it is important that the user only reveals the exact commitment that is required for the request threshold inequality proof. The original HashWires paper achieves this using an accumulator, but it is also possible to rely on the selective disclosure capabilities of SD-JWT and MSO. For reasons of readability, illustrative examples will be done using SD-JWT and without an accumulator, but the concept is equally applicable for MSO and every other salted attribute digest based approach.

NOTE: Combining HashWires range proofs with selectively disclosed salted hashes of attributes is suggested by Peter Lee Altmann (Swedish Digitalization Agency) and Sebastian Elfors (IDnow) to the present ETSI technical report. The idea is not peer reviewed and is meant primarily to illustrate the idea of an PID/(Q)EAA Provider signing computational inputs and parameters to enable dynamic predicates e.g. inequality tests. With modifications, the proposal could enhance the ISO mDL MSO [i.87] and IETF SD-JWT [i.76] standards to cater for predicate proofs in addition to selectively disclosing claims.

Consider an optimized HashWire for an *n*-digit number, $HW = \{[c_n, c_{n-1}, \ldots, c_0], [r_n, r_{n-1}, \ldots, r_0]\}$ where c_i denotes the hash chain root for digit position i in each MDP partition for a value *x* and r_i denotes the seed used in $G(\cdot)$ to generate the first value of the hash chain for each digit position *i*. Each MDP partition is a combination of hash roots.

For instance, the $MDP(6575) = \{6575,6569,6499,5999\}$ would require four seeds, resulting in four hash chains, one for each digit. The corresponding hash chains lengths for $MDP(6575)$ are $6 \cdot 10^3 + 9 \cdot 10^2 + 9 \cdot 10^1 + 9$. More precisely:

- 6575 requires the commitment: $H^6(G(r_3)), H^5(G(r_2)), H^7(G(r_1)), H^5(G(r_0))$

- 6569 requires the commitment: $H^6(G(r_3)), H^5(G(r_2)), H^6(G(r_1)), H^9(G(r_0))$
- 6499 requires the commitment: $H^6(G(r_3)), H^4(G(r_2)), H^9(G(r_1)), H^9(G(r_0))$
- 5999 requires the commitment: $H^5(G(r_3)), H^9(G(r_2)), H^9(G(r_1)), H^9(G(r_0))$

Each commitment is required to be included in a disclosure, and then signed as part of the SD-JWT or MSO. The PID/(Q)EAA Provider is required to also include a number of decoy digests to hide the number of MDP partitions, or alternatively commit only an accumulator value (e.g. a Merkle Tree as proposed in the original HashWires paper or the digest over the concatenation of all the decoys and commitments). In Figure 8, and in the example below, the commitments are included as separate disclosures for illustrative purposes only.

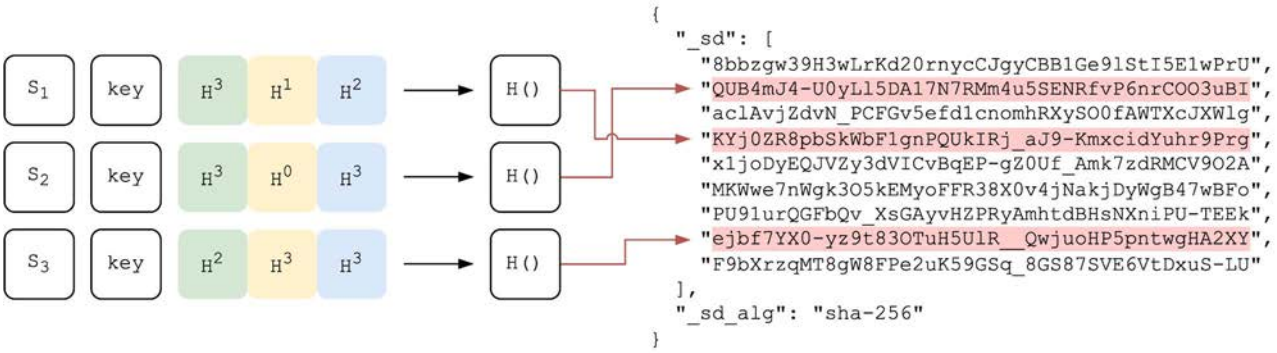

Figure 9: Optimized HashWires commitment using SD-JWT

Figure 9 illustrates an optimized HashWires commitment to the number 312 in base 4 protected by the `_sd` object suitable for an SD-JWT. Each commitment to the three partitions are salted (box with S), contain a MDP partition identifier key, and the hash chain roots for each MDP partition. The hash over the salt, key, and commitment is included in the `_sd` (red highlights). The other digests in the `_sd` object are decoys to hide the number of MDP partitions the user has. Each commitment is included as a disclosable value for illustrative purposes. Optionally, an issuer could instead add the commitments to an accumulator, which would be disclosable. This is an illustration of HashWires, although implementations may differ.

EXAMPLE: The random values needed to initiate each hash chain with $G(\cdot)$. The values are not sent to the verifier.

```
{
  "10^0": "f6a23b90b9f07f34f33dfd4e5de87adab167b6ea9eb060163e741ac26f16edc1",
  "10^1": "3026950fd2d2c6c7e23c8a8b0a80928d5cdac0f953699a96e02c1033379ed392",
  "10^2": "d942fdb1d9c3274a257154ef2f6f66161ea5872163dbb8daa40c7496e5365242",
  "10^3": "ba0acaf18a6a966a3eecbb791e9e22bc45d3a1183ff47342ab9cbde4635a828c",
  "10^4": "f32da5b457d45e0e6113d744fff316a1882f77fbf6ef5f92456faf84dfc8bd02"
}
```

The disclosure of the commitment to the partition 13699 using the format `["salt", "key", <value>]`.

```
["TpPrKdZ73ZR7JoUU-FCiTYvlQ4-QQ5ab9V2Z-cXze8E", "0",
["927eb07e71c648f73bec94e03d29cb41a0efc4f247a999d49f1318e3e8afbb84",
"b4b2a297499d63dd1ae5ee64c1aa21667b43b8974be3b3e17273005951413a56",
"854983f72c56c0102cac32edcce8b7c52365edc793cdba37d5603221b21d0a95",
"040be38408070da03bd6ca9e63999fac072adc20e1ba6f4513861db317a82a54",
"ad1a9492c27be7d33c7d00e33b0ca223e02a07440394b4036ded6f1f2c990c7a"]]
```

The base64url encoded SHA256 digest included in the `_sd`:

```
"zDHz3CX-akEjrDddMc8RYemeUCmEN0yjT1JIM_KXJd4"
```

NOTE 1: The user is required to only disclose the particular partition it uses to generate the inequality proof.

NOTE 2: The issuer can combine the disclosure digests into a single value using an accumulator or by concatenating the disclosure digests and the decoys. Implementation specific profiles are required.

The user, given a threshold value *t*, is required to select the partition that can generate the hash chains required for the inequality $x \geq t$. The user sends the disclosure of the commitment required for the inequality test, and the threshold values for each digit. The verifier can compute the hash chain using the threshold value for each digit and compares the root hash with the issuer signed commitments in the SD-JWT or MSO. If the signature is verified, the verifier accepts the inequality test.

4.5.2.5 Less than or equal to and range proofs

Any range proof, $a \leq x \leq b$, can be constructed using two inequality tests, one proving the inequality at the lower bound and the other at the upper bound. The above demonstrates an inequality test of type $a \leq x$. To generate a less than or equal to $x \leq b$ proof, it is necessary to extend the above described approach. Using whole number *K*, the issuer can generate a commitment to the inequality $K - x \geq K - b$. Both inequality tests rely solely on hash digests and combined they can generate any valid range proof using issuer signed commitments.

Example

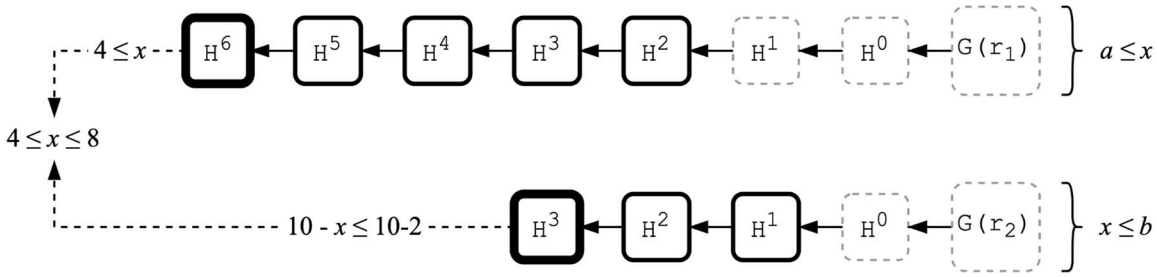

Figure 10: Hash chain based range proof

Figure 10 illustrates a hash chain based range proof for the range $4 \leq x \leq 8$. The issuer signs the bold commitments to both the lower bound test $4 \leq x$ and the upper bound test $x \leq 8$. The user presents both inequality tests to the verifier. The verifier combines the two proofs for inequality tests into range proof and accepts the range proof if the issuer's signature over the commitments is valid.

NOTE 1: For a range proof, the issuer is required to sign the parameter *K* used for the inequality $K - x \geq K - b$.

NOTE 2: The attestation issuance date impacts the proof that the user generates. A user generates a proof on an inequality test not for the request threshold, *t*, but subtracts the difference between the issuance date and the presentation date. A similar logic applies for age under or equal to proofs, as well as for range proofs.

HashWires represent an efficient way to generate inequality tests and range proofs using only SHA256. Running 70 000 loops on a dual core 2,2 GHz processor, it takes 72 µs ± 5,58 µs to generate the commitment for a 3 digit inequality test, and 156 µs ± 31,7 µs for a 6 digit one. The proof size is constant and the verification is faster than the generation.

4.5.3 zk-SNARK

4.5.3.1 Introduction to zk-SNARK

The abbreviation zk-SNARK stands for "Zero-Knowledge Succinct Non-interactive ARgument of Knowledge", and is a collaborative term for a specific category of Zero-Knowledge Proof protocols. At the time of writing (in May 2023), eighteen zk-SNARK protocols have been published by cryptographic researchers; see Annex A.4 for a list of all zk-SNARK protocols.

The zk-SNARK characteristics can be broken down as follows (based on the initials S-N-ARK) to cater for zero-knowledge (zk):

- Succinct: the proof is independent of the statement's size.
- Non-interactive: randomness is not provided by the verifier (but by a random oracle).
- ARgument of Knowledge: a proof system that demonstrates the user's knowledge of data (not just its existence).

NOTE: A zk-SNARK system provides predicate proofs, selective disclosure and unlinkability by design.

The concept of zk-SNARK was initially described by Alessandro Chiesa et al in a paper [i.31] in 2012, which in turn was based on Jens Groth's work [i.63] from 2010. The first general zk-SNARK protocol Pinocchio [i.108] was designed and implemented in 2013. In 2019, PLONK [i.59] was introduced as the universal zk-SNARK protocol.

A zk-SNARK protocol can be based on a trusted setup or as a transparent setup, as further described in clauses 4.5.3.2 and 4.5.3.3.

4.5.3.2 Trusted setup of zk-SNARK

The trusted setup of a zk-SNARK by using a witness consists of three algorithms *KeyGen*, *CP*, *CV* as illustrated in Figure 11.

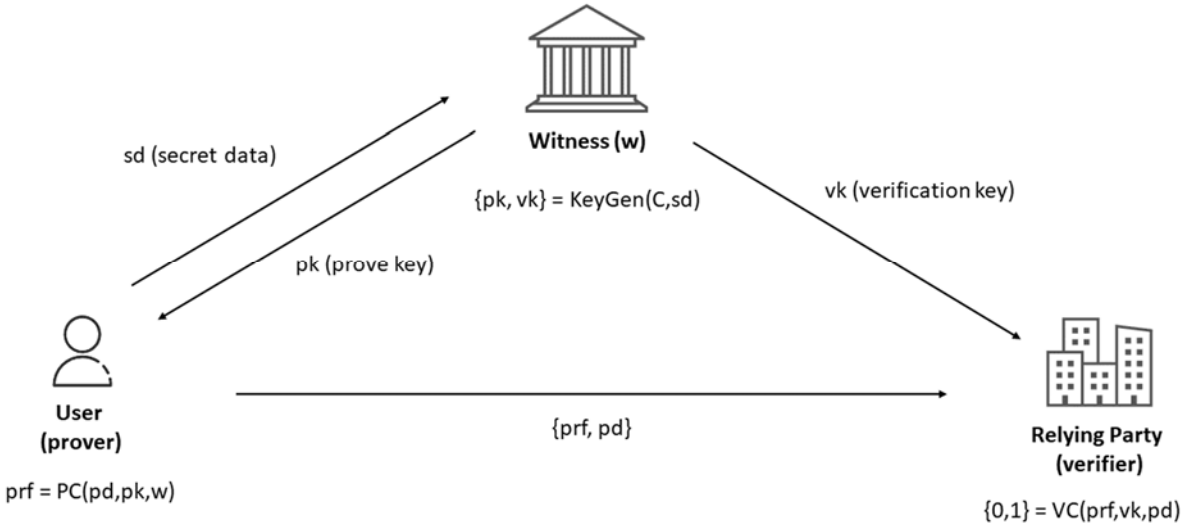

Figure 11: Overview of zk-SNARK with trusted setup

The key generator *KeyGen* takes a secret parameter *sd (secret data)* and the program *C*, and generates two publicly available keys, the user's proving key *pk*, and the relying party's verification key *vk*. These keys are public parameters that need to be generated once for a specific program *C*.

NOTE 1: The parameter *sd* used in the generator is a secret value. If this parameter is known to an attacker, it can generate a set of fraudulent proving key and verification key, which can generate fake proofs without knowing the witness *w*.

NOTE 2: An eIDAS2 QTSP could serve the role as a witness, since it is aware of the user's data in the first place.

EXAMPLE 1: The secret parameter *sd* could be the user's real age, for example 21.

The user executes the algorithm *CP* with the following input parameters: its proving key *pk*, a public input *pd (public data)*, and a private witness *w*. The algorithm *CP* generates the proof value *prf = CP(pk, pd, w)*, as evidence that the user knows a witness *w*.

EXAMPLE 2: The public data *pd* could be the statement, for example that the user's age is above 18.

The verifying relying party calculates the algorithm *CV(vk, pd, prf)* which returns true if the proof is correct and false otherwise. Hence, the function *CV* returns true if the user knows a witness *w* that satisfies the function *C(sd,w) = true*.

EXAMPLE 3: zk-SNARK protocols with trusted setup are Pinocchio [i.108], Geppetto [i.35], and TinyRAM [i.8]. For a complete list of zk-SNARK protocols with trusted setups, see table A.4 in clause A.4.

4.5.3.3 Transparent setup of zk-SNARK

In a transparent (public) setup of zk-SNARK there is no need for a trusted setup with a witness. As a tradeoff, the performance of a transparent zk-SNARK protocol is lower than a zk-SNARK with trusted setup.

EXAMPLE: zk-SNARK protocols with transparent (public) setups are SuperSonic [i.97], Hyrax [i.120] and Halo [i.14]. For a complete list of zk-SNARK protocols with transparent setups, see table A.4 in clause A.4.

4.5.3.4 Cryptography behind zk-SNARK

The cryptography that underpin the zk-SNARK schemes is highly complex and differs from protocol to protocol.

In brief, the zk-SNARK protocols can be constructed based on the following categories of cryptographic frameworks [i.109]:

- Fiat-Shamir Heuristics, which in turn can be broken down into Sigma-Protocols, Random Oracle Models (ROM) and Fiat-Shamir-Compatible Hash Functions.
- Probabilistically Checkable Proofs (PCP): Merkle Trees and Hash Functions, Kilian Interactive Argument of Knowledge, and Micali's Computationally Sound (CS) Proof.
- Quadratic Arithmetic Programs (QAPs) and Square Span Programs (SSPs).
- Linear Interactive Proofs (LIPs).
- Polynomial Interactive Oracle Proofs (PIOPs).

Given the vast literature of zk-SNARK algorithms, a complete description of the cryptography for zk-SNARK goes beyond the scope of the present document. For further reading about the cryptographic algorithms behind the zk-SNARK protocols, the following papers are recommended: Nitulescu "zk-SNARKs: A Gentle Introduction" [i.102] and Petkus "Why and How zk-SNARK Works: Definitive Explanation" [i.109].

4.5.3.5 Implementations

As regards to implementations, zk-SNARK was implemented in 2016 for the blockchain protocol ZeroCash for cryptocurrency ZCash, for which zk-SNARK caters for four different transaction types: private, shielding, deshielding, and public. Hence, zk-SNARK allows the users to determine how much data to be shared with the public ledger for each transaction. The blockchain Ethereum zk-Rollups also utilizes zk-SNARKs to increase its scalability.

4.5.3.6 Cryptographic analysis

Whether a zk-SNARK protocol is quantum-safe or not depends on the underlying cryptographic algorithms, as described in table A.4. The zk-SNARK protocols Aurora [i.9], Ligero [i.3], Spartan [i.99], and Virgo [i.132] are considered as plausible quantum-safe, whilst the others in table A.4 are not considered as quantum-safe.

4.5.4 zk-STARK

4.5.4.1 Introduction to zk-STARK

The abbreviation zk-STARK stands for "Zero-Knowledge Succinct Transparent Arguments of Knowledge", and is a collaborative term for a specific category of Zero-Knowledge Proof protocols. The zk-STARK protocols fulfill the criteria of a Zero-Knowledge Proof system, which enables one party (the prover) to prove to another party (the verifier) that a certain statement is true, without revealing any additional information beyond the truth of the statement itself. Furthermore, zk-STARKs are succinct, such that they allow for the creation of short proofs that are easy to verify, and they are transparent, meaning that anyone can verify the proof without needing any secret information.

The zk-STARK characteristics can be broken down as follows (based on the initials S-T-ARK) to cater for zero-knowledge (zk):

- Scalable: the prover algorithm is typically implemented with repeated functions (e.g. several hash functions).
- Transparent: the prover and verifier keys are generated verifiably in a trustless manner (i.e. without the need of a trusted setup).
- ARgument of Knowledge: a proof system that demonstrates the user's knowledge of data (not just its existence).

NOTE: A zk-STARK system provides predicate proofs, selective disclosure and unlinkability by design.

The concept of zk-STARK was initially described by Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev in a paper [i.7], 2018. At the time of writing (in May 2023), two zk-STARK protocols have been published by cryptographic researchers:

- the zk-STARK protocol [i.6] in 2019; and
- Zilch [i.101] in 2021.

4.5.4.2 Setup of zk-STARK

Unlike the zk-SNARK frameworks, which in several cases require a trusted setup, the zk-STARK protocols are designed to be used without a trusted setup. Hence, the zk-STARK protocols are considered to be both transparent and universal: a transparent protocol is defined as it does not require any trusted setup and uses public randomness, and a universal protocol is defined as it does not require a separate trusted setup for each circuit.

4.5.4.3 Cryptography behind zk-STARK

The cryptography behind the zk-STARK schemes is based on Interactive Oracle Proofs (IOP) with scalable proofs.

A Zero-Knowledge system based on IOP (ZK-IOP) [i.6] is a common generalization of the Interactive Proofs (IP), Probabilistically Checkable Proofs (PCP) and Interactive PCP (IPCP) models that were previously introduced for zk-SNARKs (see clause 4.5.3).

The zk-STARK protocols are typically implemented using standard hash functions. As in the PCP model, the IOP verifier does not need to read all prover messages, but can rather query them at random locations; as in the IP model, prover and verifier interact over several rounds. Hence, a ZK-IOP system can be converted into an interactive ARgument of Knowledge (ARK) model, assuming a family of collision-resistant hash functions can be turned into a non-interactive argument in the random oracle model, which is typically realized using a standard hash function.

Given the complexity of zk-STARK algorithms, a complete description of the cryptography for zk-STARK goes beyond the scope of the present document. For further reading about the cryptographic algorithms behind the zk-STARK framework, the following paper is recommended: Ben-Sasson et al "Scalable, transparent, and quantum-safe computational integrity" [i.6].

4.5.4.4 Implementations

Potentially, zk-STARKs could replace zk-SNARKs for various applications in the future. For example, zk-STARKs could be used for the privacy and confidentiality of ZeroCash protocol, which is currently implemented with zk-SNARK. However, zk-SNARKs are roughly 1000 times shorter than zk-STARK proofs, so replacing zk-SNARKs with zk-STARKs would require more research to either shorten proof length, or aggregate and compress several zk-STARK proofs using incrementally verifiable computation [i.6].

4.5.4.5 Cryptographic analysis

The zk-STARK schemes are considered as plausible quantum-safe, since they are based on a machinery of hash functions for implementing the IOP. If the used hash functions are designed as QSC, the zk-STARK scheme becomes quantum-safe.

5 Credential formats with selective disclosure

5.1 General

The present clause provides an analysis of a set of formats for selective disclosure.

The topics for the analysis of each selective disclosure credential formats are:

- Signature scheme(s) used for selective disclosure and optionally Zero-Knowledge Proofs, when applicable with references to clause 4.
- Encoding of the credentials used for selective disclosure.
- Maturity of the credential format's specification and deployment.
- Cryptographic aspects, more specifically if the cryptographic algorithms used for the selective disclosure credential formats are approved by SOG-IS and allows for QSC algorithms for future use.

This credential formats are categorized according to three of the main cryptographic schemes for selective disclosure:

- Atomic (Q)EAA credential formats, see clause 5.2. These credential formats correspond to the (Q)EAA signature schemes described in clause 4.2.
- Multi-message signature credential formats, see clause 5.3. These credential formats correspond to the multi-message signature schemes described in clause 4.3.
- Credentials with hashes of salted attributes, see clause 5.4. These credential formats correspond to the multi-message signature schemes described in clause 4.4.

NOTE 1: There is also a type of generic JSON container format (JSON WebProofs), which allows for a mix of the selective disclosure signature schemes in clause 4, and is therefore treated as a separate category of credential formats.

NOTE 2: The proofs for arithmetic circuits do not rely upon credential formats per se, but are rather schemes for ZKP. Hence, proofs for arithmetic circuits are out of scope for this clause, which describes credential formats.

5.2 Atomic (Q)EAA credential formats

5.2.1 Introduction to atomic (Q)EAA formats

The concept of atomic (Q)EAAs was introduced in clause 4.2. There are numerous credential formats that can be issued with a single claim, so in principle a selective disclosure scheme based on atomic claims can be designed for a variety of types of credentials formats (ICAO DTCs, IETF JWTs, W3C Verifiable Credentials, X.509 certificates, etc.).

Clauses 5.2.2 and 5.2.3 are however focusing in more detail on two (Q)EAA formats that are used for atomic (Q)EAA schemes: PKIX X.509 attribute certificates and W3C Verifiable Credentials.

5.2.2 PKIX X.509 attribute certificate with atomic attribute

The PKIX X.509 Attribute Certificate (AC) profile is specified in IETF RFC 5755 [i.78]. An attribute certificate may contain attributes that specify group membership, role, security clearance, or other authorization attributes associated with the user. The attribute certificate is a signed set of attributes, although it does not contain a public key. Instead, the attribute certificate is linked to a X.509 Public Key Certificate (PKC), which can be used by the user for authentication. In order to preserve the user's privacy, the X.509 public key certificate may only include a pseudonym in the subject field.

The attribute certificates are issued by an Attribute Authority (AA), and they may be issued with a short lifetime and with an atomic (single) attribute. These characteristics make short-lived attribute certificates with atomic credentials suitable for an access control service with selective disclosure features.

A description of how to use PKIX X.509 attribute certificates for selective disclosure with an access control system is available in clause 6.2.1.

The X.509 attribute certificates are ASN.1/DER encoded as described in IETF RFC 5755 [i.78].

X.509 certificates can be signed by the QTSP using cryptographic algorithms (RSA with proper key lengths or ECC with approved curves) that are published by SOG-IS [i.115]. For future use, the X.509 certificates can be signed with quantum-safe cryptographic algorithms [i.93].

The maturity of X.509 attribute certificates can be considered as high, given that the IETF RFC 5755 [i.78] is a mature PKIX standard.

5.2.3 W3C Verifiable Credential with atomic attribute

As a preparation for enrollment of W3C Verifiable Credentials with atomic attributes, the EUDI Wallet would need to be equipped with Credential templates for the W3C Verifiable Credentials. The W3C Verifiable Credentials Data Model v1.1 [i.128] distinguishes between a Credential as "a set of one or more claims made by an issuer" and a Verifiable Credential as "a verifiable credential is a tamper-evident credential that has authorship that can be cryptographically verified". Put differently, a Verifiable Credential can be a signed Credential. Hence, the Credential(s) in the EUDI Wallet can consist of templates with the attribute properties that should be used for the enrollment of attribute values.

NOTE: The W3C Verifiable Credentials Data Model v1.1 [i.128] is a conceptual data model rather than a specific credential format. In this context of atomic attributes, however, the scope of W3C Verifiable Credentials can be limited to the JWT format.

A description of how to use the FIDO standard as an authentication protocol in conjunction with Verifiable Credentials with atomic attributes for selective disclosure is available in clause 6.2.2.

The encoding of the W3C Verifiable Credentials is specified as JWT or JSON-LD in the W3C Verifiable Credentials Data Model v1.1 [i.128].

W3C Verifiable Credentials can be signed by the QTSP using cryptographic algorithms (RSA with proper key lengths or ECC with approved curves) that are published by SOG-IS [i.115]. For future use, the W3C Verifiable Credentials can be signed with quantum-safe cryptographic algorithms as described in the IETF report on JOSE signatures with QSC algorithms [i.73].

The maturity of W3C Verifiable Credentials can be considered as high, given the wide deployment of issued W3C Verifiable Credentials.

5.3 Multi-message signature credential formats

5.3.1 W3C VC Data Model with ZKP

The W3C Verifiable Credentials (VC) Data Model v1.1 [i.128] contains clause "5.8 Zero-Knowledge Proofs", which describes a data model that supports selective disclosure with the use of Zero-Knowledge Proof (ZKP) mechanisms.

The W3C Verifiable Credentials Data Model states two requirements for Verifiable Credentials when they are to be used in ZKP systems:

- The Verifiable Credential contains a proof, so that the user can derive a verifiable presentation that reveals only the information that the holder intends to reveal.
- The credential definition (if being used) is defined in the JSON credentialSchema property, so that it can be used to perform various cryptographic operations in zero-knowledge.

The following cryptographic schemes that support selective disclosure while protecting privacy across multiple presentations have been implemented for the W3C Verifiable Credentials Data Model [i.128]: BBS [i.64], CL Signatures [i.23], Idemix [i.69], Merkle Disclosure Proof 2021 [i.123], Mercurial Signatures [i.37], PS Signatures [i.110], U-Prove [i.1] and Spartan [i.114].

More specifically, the W3C Verifiable Credentials Data Model standard includes examples of how to use Camenisch-Lysyanskaya (CL) signatures (see clause 4.3.2) with a W3C Verifiable Credential and a W3C Verifiable Presentation; see examples 24 and 25 in W3C Verifiable Credentials Data Model [i.128] for examples of these data structures.

An example of how to combine two W3C Verifiable Credentials into a W3C Verifiable Presentation with selected attributes is shown in Figure 12.

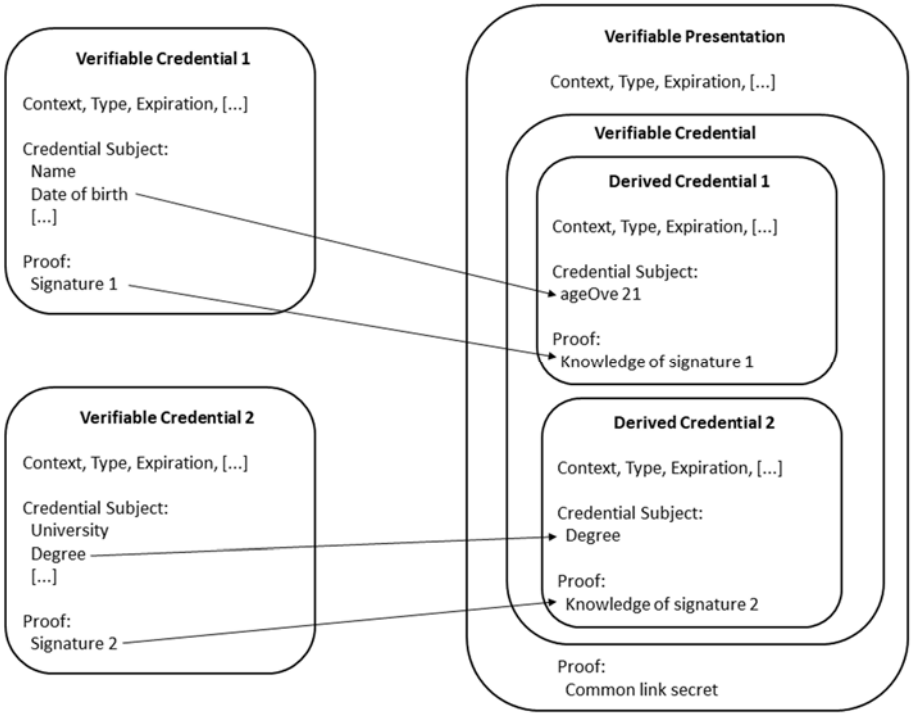

Figure 12: W3C Verifiable Credentials presented using ZKP

In Figure 12, selectively disclosed attributes from W3C Verifiable Credential 1 and W3C Verifiable Credential 2 are combined into a W3C Verifiable Presentation. CL-signatures are used in the Verifiable Presentation to create the proofs of knowledge of the original W3C Verifiable Credential signatures.

5.3.2 W3C VC Data Integrity with BBS Cryptosuite

W3C BBS Cryptosuite v2023 [i.130] is an experimental draft specification, which defines a set of cryptographic suites for the purpose of creating, verifying and deriving proofs for the BBS signature scheme (see clause 4.3.1). The BBS signatures are compatible with any pairing friendly elliptic curve, however the cryptographic suites defined in the BBS Cryptosuite specification allow the usage of the BLS12-381 curve for interoperability purposes.

W3C BBS Cryptosuite v2023 [i.130] can be used in conformance with the W3C Verifiable Credentials Data Integrity specification [i.127], which in turn describes mechanisms for ensuring the authenticity and integrity of JSON-LD encoded W3C Verifiable Credentials, especially through the use of digital signatures and related cryptographic proofs.

As a result, the BBS signature scheme (clause 4.3.1) can be applied on W3C Verifiable Credentials and W3C Verifiable Presentations in order to disclose selected attributes, which are signed by the user's proofs without revealing the entire W3C Verifiable Credentials and their original signatures.

5.3.3 Hyperledger AnonCreds (credential format)

The Hyperledger AnonCreds [i.64] credentials are JSON-formatted according to public AnonCreds objects, which in turn are defined by Schemas, CredDefs, Revocation Registry Definitions and Rev_Reg_Entrys. These objects are published by the issuers to repositories called Verifiable Data Registries (VDRs), which are accessible to users and verifiers to enable presentation generation and verification. AnonCreds can also be issued in accordance with the W3C Verifiable Credentials Data Model.

AnonCreds are bound to the user with a non-correlatable secret only known to the user itself called a link secret. The link secret as a blind attribute that is sent to the issuer during credential issuance. The issuer signs every claim (including the blinded link secret) individually, enabling selective disclosure. The Pedersen Commitment is used for the link secret. It means the issuer does not know the exact value of the link secret, and the holder can prove the ownership of credentials to a verifier without disclosing a persistent identifier.

The cryptographic signature scheme used by AnonCreds is CLRSA-signatures (see clause 4.3.2), which caters for selective disclosure and Zero-Knowledge Proofs.

More information about the AnonCreds protocols is available in clause 6.3.

5.3.4 Cryptographic analysis

The maturity of W3C Verifiable Credentials can be considered as high, given the wide deployment of issued W3C Verifiable Credentials. However, BBS and CL signatures are not secure against quantum-safe cryptographic algorithms [i.118], and they are additionally not standardized by NIST in the US or by SOG-IS in the EU. Furthermore, since AnonCreds are based on CLRSA-signatures, the cryptographic algorithms are not considered as quantum-safe nor SOG-IS approved.

5.4 Credentials with hashes of salted attributes

5.4.1 General

The general concept of selective disclosure based on hashes of salted attributes is described in clause 4.2. As regards to credentials within this category, there are two main formats:

- IETF SD-JWT, which is further described in clause 5.4.2.
- ISO mDL MSO (Mobile Security Object), which is elaborated in clause 5.4.3.

NOTE: ETSI EN 319 162-1 [i.42] specifies the Associated Signature Containers (ASiC), which is an XML-formatted manifest that binds together a number of hashed file objects into one single digital container. The principle of combining hashed objects in an ASiC manifest is similar to the IETF SD-JWT and ISO mDL MSO credentials with hashes of salted attributes. There are however two main differences:

- ETSI ASiC is intended for combining file objects in a signature container manifest, whilst IETF SD-JWT and ISO mDL MSO are designed for selective disclosure.

Furthermore, the ETSI ASiC hashes are not salted, whilst the hashed attributes in IETF SD-JWT and ISO mDL MSO are salted to cater for unlinkability. Hence, the comparison with ETSI ASiC is observed, but nevertheless out of scope for this clause.

5.4.2 IETF SD-JWT

To support selective disclosure in JWTs, IETF has specified Selective Disclosure JSON Web Token (SD-JWT) [i.76]. At its core, an SD-JWT is a digitally signed JSON document that can contain salted attribute digests that the user can selectively disclose using disclosures that are outside the SD-JWT document. This allows the user to share only those attributes that are strictly necessary for a particular service. The technique of SD-JWT is based on hashed salted attributes as described in clause 4.4.

Each SD-JWT contains a header, payload, and signature. The header contains metadata about the token including the type and the signing algorithm used. The signature is generated using the issuer's private key. The payload includes the proof object that enables the selective disclosure of attributes. Each disclosure contains a salt, a cleartext claim name, and a cleartext claim value. The issuer then computes the hash digest of each disclosure and includes each digest in the attestation it signs and issues.

NOTE: The JOSE [i.84] signature format allows for SOG-IS approved cryptographic algorithms [i.115] and QSC algorithms [i.73] for future use.

The SD-JWT specification is still a draft, yet SD-JWT has been selected in the ARF [i.34] as the JSON-format for selective disclosure.

A thorough analysis of SD-JWT and how it can be applied for selective disclosure of the PID/(Q)EAA for the EUDI Wallet is available in clause 7.3.

5.4.3 ISO/IEC 18013-5 Mobile Security Object (MSO)

The Mobile Security Object (MSO) is specified in clause 9.1.2.4 of ISO/IEC 18013-5 [i.87] and contains the following attributes encoded in a CDDL [i.85] structure:

- digestAlgorithm: Message digest algorithm
- valueDigests: Array of digests of all data elements
- deviceKey: Device key in COSE_Key as defined in IETF RFC 8152 [i.83]
- docType: DocType as used in Documents
- validityInfo: validity of the MSO and its signature

The valueDigests are issued as IssuerSignedItems, which are the hash values of the ISO mDL attributes combined with random values (see ISO/IEC 18013-5 [i.87], clause 9.1.2.4). In other words, the MSO is a selective disclosure standard based on salted hashes of attributes (see clause 4.4), where the random values are the salts.

The deviceKey contains the mDL Authentication Key (see clause 7.2.2), which is protected by the user's PIN-code or biometrics (see clause 7.4).

The MSO is signed by the mDL Issuer Authority, which is an IACA X.509 CA (see clause 7.2.1.4), and the signature is COSE formatted.

NOTE 1: ISO/IEC 18013-5 [i.87] "Table B.3 - Document signer certificate" lists the ECDSA curves BrainpoolP256r1, BrainpoolP384r1 and BrainpoolP512r1, which are also approved by SOG-IS [i.115].

NOTE 2: The COSE [i.79] signature format also allows for QSC algorithms [i.72] for future use.

An example of an ISO mDL MSO data structure is provided in ISO/IEC 18013-5 [i.87], annex D.5.2.

The MSO is stored and protected in the device's SE/TEE. The MSO is included in the mDL Response for the device retrieval flow (see clause 7.2.3).

ISO/IEC 18013-5 [i.87] is considered mature, and several ISO mDL device retrieval solutions with MSOs have been deployed in production, for example in a number of states in the US.

A thorough analysis of ISO mDL MSO and how it can be applied for selective disclosure of the PID/(Q)EAA for the EUDI Wallet is available in clause 7.2.

5.5 JSON container formats

5.5.1 IETF JSON WebProof (JWP)

The JOSE [i.74] standard is a widely adopted container format for JSON-formatted Keys (JWK), Signatures (JWS), and Encryption (JWE). For example, JWTs with JOSE-containers are used by the OpenID Connect standard and by W3C's Verifiable Credentials.

However, JOSE is not designed to cater for the growing number of Zero-Knowledge Proof and selective disclosure schemes. Most of these emerging cryptographic schemes require additional transforms, are designed to operate on subsets of messages, and have more input parameters than traditional signature algorithms.

Examples of selective disclosure signature schemes that would benefit from a more flexible JSON container format are:

- BBS [i.71];
- CL Signatures [i.23];
- Idemix [i.62];

- Merkle Disclosure Proof 2021 [i.123];
- Mercurial Signatures [i.37];
- PS Signatures [i.110];
- U-Prove [i.1]; and
- Spartan [i.114].

They adhere to the same principles of collecting multiple attributes and binding them together into a single issued token, which is transformed into a presentation that reveals only a subset of the original attributes, predicate proofs, or proofs of knowledge of the attribute.

In order to address these issues, the IETF JSON working group has drafted the JSON WebProof (JWP) specification. The JWP specification defines a new JSON container format similar in design to JSON Web Signature (JWS). However, JWS only integrity-protects a single payload, whilst JWP can integrity-protect multiple payloads in one message. JWP also specifies a new presentation form that supports selective disclosure of individual payloads, enables additional proof computation, and adds a protected header to prevent replay and support binding mechanisms.

5.5.2 W3C JSON Web Proofs For Binary Merkle Trees

In hash-based cryptography, the Merkle signature scheme is a digital signature scheme based on Merkle trees and one-time signatures such as the Lamport signature scheme. It was developed by Ralph Merkle in the late 1970s and is an alternative to traditional digital signatures such as DSA or RSA. An advantage of the Merkle signature scheme is that it is plausible quantum-safe.

The JSON Web Proofs For Binary Merkle Trees [i.122] specification defines a generic encoding of merkle audit paths that is suitable for combining with JWS to construct selective disclosure proofs. The specification is suitable for more generic applications and formats such as W3C Verifiable Credentials [i.128] and W3C Decentralized Identifiers [i.121].

JSON Web Proofs (see clause 5.5.1) are used as formats for the encoding binary merkle trees.

Selective disclosure is defined as the same as full disclosure with the exception that the rootNonce is not encoded in the compressed representation. The rootNonce is omitted in order to ensure that a selective disclosure proof does not reveal information that can be used to brute force siblings of disclosed members.

Merkle proofs are already being used to provide certificate transparency in IETF RFC 9162 [i.86]. The JSON Web Proofs For Binary Merkle Trees [i.122] specification is however independent of the certificate transparency specification.

6 Selective disclosure systems and protocols

6.1 General

The present clause provides an analysis of a set of systems and protocols for selective disclosure.

The topics for the analysis of each selective disclosure protocol are:

- Signature scheme(s) used for selective disclosure and optionally Zero-Knowledge Proofs, when applicable with references to clause 4.
- Credential format(s) for selective disclosure, when applicable with references to clause 5.
- Protocol(s) for presentation of the user's credentials to a relying party (relying party).
- Maturity of the protocol's specification and deployment.
- Cryptographic aspects, more specifically if the cryptographic algorithms used for the selective disclosure protocol are approved by SOG-IS and allows for QSC algorithms for future use.

The selective disclosure systems and protocols described in this clause cater for a mix of selective disclosure credential formats and signature schemes, so they are not categorized in the same way as in clauses 4 and 5.

6.2 Atomic attribute credential presentation protocols

6.2.1 PKIX X.509 attribute certificates with single attributes

An access control system based on PKIX X.509 certificates with atomic attributes is illustrated in Figure 13.

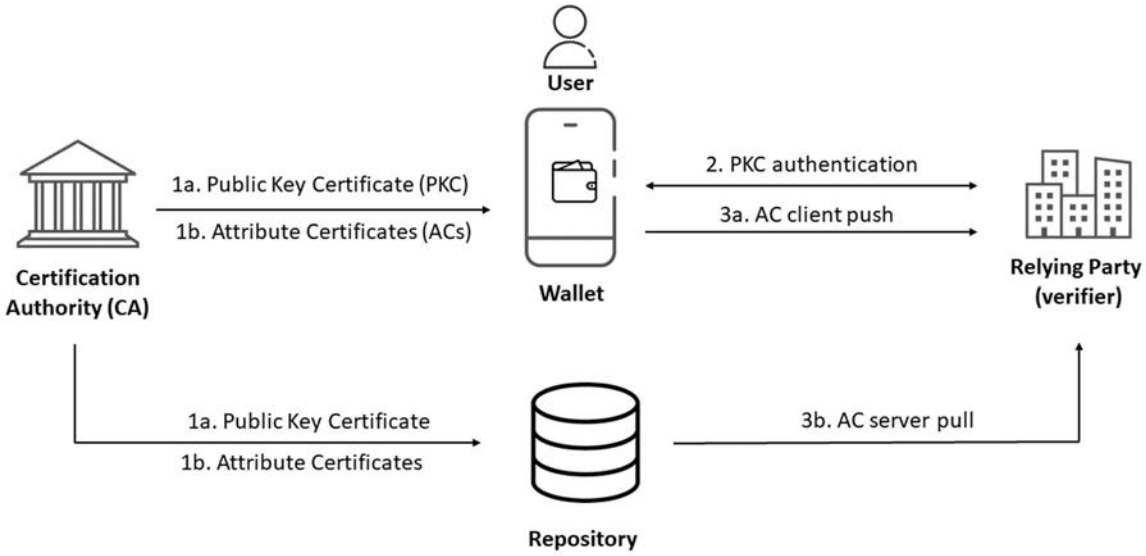

Figure 13: Overview of attribute certificate authorization

First, the system is configured by a Certification Authority (CA) that issues a PKIX X.509 public key certificate to a user's wallet. The user has a corresponding private key protected in the wallet, such that the user can be authenticated with the public key certificate. The public key certificate may only contain a pseudonym. The Certification Authority also issues short-lived PKIX X.509 attribute certificates with atomic attributes. The attribute certificates are associated with the public key certificate, and they may be stored in the user's wallet and/or in a central repository.

Second, the user authenticates to a relying party (with an access control system) by using the public key certificate. For example, TLS/SSL could be used for this authentication. If the public key certificate only contains a pseudonym of the user, the authentication protocol does not reveal the user's identity.

Third, the user's attribute certificate(s) are submitted to the relying party's access control system. The attribute certificate(s) may either be pushed from the client to the relying party, or pulled from the repository by the relying party.

For more information about attribute certificate architectures, see the IETF RFC 5755 [i.78].

An alternative design of using attribute certificates for anonymous authorization is described in the paper "A First Approach to Provide Anonymity in Attribute Certificates" [i.11] from 2004.

The PKIX X.509 certificates can be signed with SOG-IS approved cryptographic algorithms and allows for QSC algorithms for future use, meaning that the attribute certificate access control solution meets the SOG-IS requirements on cryptographic algorithms.

6.2.2 VC-FIDO for atomic credentials

Another example of a protocol for selective disclosure based on atomic credentials is the VC-FIDO [i.26] integration that was invented at Kent University. The used atomic (Q)EAA format is W3C Verifiable Credential, which is described in clause 5.2.3.

In order to issue the atomic W3C Verifiable Credentials to an EUDI Wallet, the user needs to be identified or authenticated to a QTSP. The VC-FIDO integration is based on the W3C WebAuthn protocol in the FIDO2 standard. The WebAuthn [i.129] stack is extended with a W3C Verifiable Credentials enrollment protocol, resulting in a client that can enroll for multiple atomic short-lived W3C Verifiable Credentials based on W3C Credential templates. These atomic short-lived W3C Verifiable Credentials can then be (temporarily) stored in an EUDI Wallet, and be combined into a Verifiable Presentation that is presented to the relying party (verifier). Selective disclosure is achieved since the user can enroll for the atomic attributes it needs for a specific use case, and present only those atomic (Q)EAAs to a Relying Party.

The VC-FIDO integration was presented by David Chadwick at SHACK2020 [i.26]. This presentation explains the VC-FIDO architecture diagrams and shows a demo of how the client enrolls for three atomic W3C Verifiable Credentials (address, driving license, and credit card) that are combined into a Verifiable Presentation as a parking ticket. The VC-FIDO integration is still a prototype, which is deployed as a pilot at National Health Services (NHS) in the UK.

The W3C Verifiable Credentials can be signed with SOG-IS approved cryptographic algorithms and allows for QSC algorithms for future use, meaning that the VC-FIDO solution meets the SOG-IS requirements on cryptographic algorithms.

6.3 Hyperledger AnonCreds (protocols)

The Hyperledger AnonCreds (Anonymous Credentials) specification [i.64] is based on the open source verifiable credential implementation of Hyperledger AnonCreds that has been in use since 2017, first as part of the Hyperledger Indy [i.67] open source project and since 2022 in the Hyperledger AnonCreds project. The Hyperledger AnonCreds credential format is described in clause 5.3.3.

The Hyperledger AnonCreds protocols can be implemented by the combination of Hyperledger Aries [i.65] protocols and Hyperledger Indy [i.67] credentials. The Hyperledger AnonCreds cryptographic support is provided by Hyperledger Ursa [i.68] for the public/private key pair management, signatures and encryption.

Hyperledger AnonCreds are widely deployed, and are for example used by organizations such as the Government of British Columbia, IDunion, and the IATA Travel Pass.

6.4 Idemix (Identity Mixer)

The Idemix (Identity Mixer) technology [i.69] was invented by IBM® Research in 2008. The Idemix system caters for strong authentication that is privacy preserving based on ABC (Attribute Based Credentials).

In summary, the Idemix scheme contains two protocols: Issuing the credential to a user and presenting it when accessing a relying party. An overview of the Idemix ABC scheme is illustrated in Figure 14.

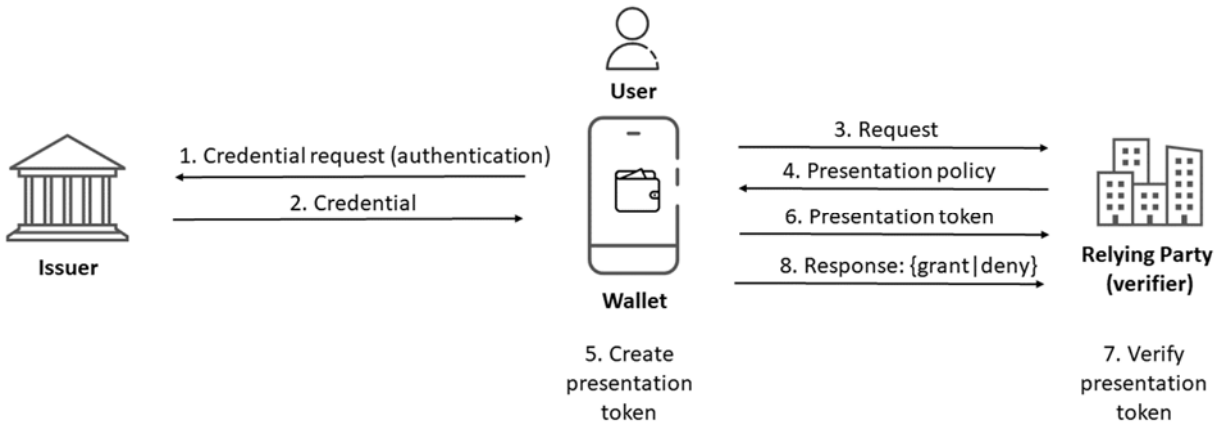

Figure 14: Overview of the Idemix ABC scheme

The Idemix system supports selective disclosure based on unlinkable Zero-Knowledge Proofs, such that users can prove that they are over 18 years old without revealing their name or birthdate. Idemix uses the pairing-based CL-signature scheme (clause 4.3.2) to prove knowledge of a signature in a Zero-Knowledge Proof.

NOTE 1: CL-signatures are not SOG-IS approved and plausible not quantum-safe.

The Idemix solution has been implemented by IBM® Identity Mixer [i.69], Hyperledger Fabric [i.66], Radboud University Nijmegen's IRMA project [i.112], and the EU-project PrimeLife [i.111]. The Idemix system was also selected as an ABC solution by the EC-funded project Attribute based Credentials for Trust (ABC4Trust) [i.70].

NOTE 2: Idemix is similar to the U-Prove (see clause 6.7) in the sense that both protocols are based on privacy-preserving ABC technology, although the iterations in the issuance phase and the underlying cryptographic algorithms differ.

The Idemix ABC system has been formalized by Camenisch et al in the paper "A Formal Model of Identity Mixer" [i.25] and the Idemix revocation mechanisms are discussed by Lapon et al in the paper "Analysis of Revocation Strategies for Anonymous Idemix Credentials" [i.96].

6.5 ISO mobile driving license (ISO mDL)

6.5.1 Introduction to ISO/IEC 18013-5 (ISO mDL)

The ISO mobile driving license (ISO mDL) is specified in the ISO/IEC 18013-5 [i.87] standard, which on a high level can be divided in the device retrieval flow (see clause 6.5.2) and the server retrieval flows (see clause 6.5.3) for selective disclosure of the user's mDL data.

ISO/IEC 18013-7 [i.88] is a draft specification that extends the ISO/IEC 18013-5 [i.87] standard with unattended flows (see clause 6.5.3), which are online protocols for selective disclosure of the user's mDL data to a web hosted ISO mDL reader.

6.5.2 ISO/IEC 18013-5 (device retrieval flow)

The ISO mDL device retrieval flow is described in ISO/IEC 18013-5 [i.87], clauses 6.3.2, 6.3.2.1 (as flow 1) and 6.3.2.4.

The credential format is the ISO mDL data, which contains the attributes about the user, in conjunction with the Mobile Security Object (MSO). The MSO is a signed object that contains a list of salted hash values of the user's mDL attributes. The MSO caters for selective disclosure based on the salted hash values as described in clause 5.4.2.

The selected attributes of the ISO mDL data and the MSO are presented by the user's ISO mDL app to an ISO mDL reader by using BLE, NFC or WiFi. The ISO mDL reader verifies the MSO and the selectively disclosed attributes (see clause 7.2.4 for more information on the ISO mDL device retrieval flow).

ISO/IEC 18013-5 [i.87] is considered mature, and several ISO mDL device retrieval solutions have been deployed in production, for example in a number of states in the US.

The ISO mDL MSO and DeviceSignedItems can be signed with cryptographic algorithms that are currently approved by SOG-IS [i.115]. Since the MSO and DeviceSignedItems are signed with a COSE-formatted signature, this caters for MSOs to be signed in the future with QSC algorithms such as CRYSTALS Dilithium [i.38], FALCON [i.57], or SPHINCS+ [i.116] as discussed in the IETF report "JOSE and COSE Encoding for Post-Quantum Signatures" [i.73].

NOTE: Although DeviceSignedItems can be signed with candidate quantum-safe signatures, the issue of having a quantum-safe key agreement mechanism to secure the communication channel remains. The ephemeral session keys between the ISO mDL device and the reader are currently exchanged using the ECKA-DH key agreement, which is vulnerable to quantum computing attacks. Furthermore, MAC signatures are mentioned in ISO/IEC 18013-5 [i.87] as offering better privacy guarantee, but the MAC secret is derived from an ECKA-DH key agreement, which is exposed to the quantum computing vulnerability. An extensive analysis of the ISO mDL session key exchange goes beyond the scope of the present document, however, but this quantum computing vulnerability should be observed.

The ISO mDL device retrieval flow has been selected as a PID protocol for EUDI Wallet Type 1 configurations in the ARF [i.34].

An extensive analysis of the ISO mDL device retrieval flow, and how it can be applied for eIDAS2 QTSPs and EUDI Wallet PID/(Q)AEE, is available in clause 7.2.3.

6.5.3 ISO/IEC 18013-5 (server retrieval flows)

The ISO mDL server retrieval flows are described in ISO/IEC 18013-5 [i.87], clause 9.2.

The ISO mDL server retrieval flow can be initialized as a hybrid device/server process (see clause 7.2.4.2) or as a server process (see clause 7.2.4.3). Once the ISO mDL server retrieval flow has been initialized, it continues with either the WebAPI flow or the OpenID Connect (OIDC) flow.

In the WebAPI flow the mDL Reader submits a server retrieval WebAPI Request with a list of requested DataElements to the Issuing Authority. Upon the user's consent, the Issuing Authority will reply with the mDL Response with the selected and disclosed DataElements (see clause 7.2.4.4 for more information).

In the OIDC flow the mDL Reader (OIDC client) submits a server retrieval OIDC Request with the requested data elements (JWT claims) to the Issuing Authority, which operates an OIDC Authorization Server. This activates the OIDC authorization code flow [i.105]. Based on the user's consent, the Issuing Authority (OIDC Authorization Server) will reply to the mDL Reader (OIDC client) with the OIDC Token with the selected and disclosed JWT claims about the user (see clause 7.2.4.5 for more information).

ISO/IEC 18013-5 [i.87] and OIDC standards are considered mature, and several ISO mDL server retrieval solutions have been deployed in production, for example in a number of states in the US.

The WebAPI and OIDC tokens are JWTs that can be signed with cryptographic algorithms that are currently approved by SOG-IS [i.115]. Since the WebAPI and OIDC tokens are signed with a JOSE-formatted signature, this caters for those JWTs to be signed in the future with QSC algorithms such as CRYSTALS Dilithium [i.38], FALCON [i.57], or SPHINCS+ [i.116] as discussed in the IETF report "JOSE and COSE Encoding for Post-Quantum Signatures" [i.73].

The ISO mDL server retrieval flows have been selected as a PID protocol for EUDI Wallet Type 1 configurations in the ARF [i.34].

An extensive analysis of the ISO mDL server retrieval flow, and how it can be applied for eIDAS2 QTSPs and EUDI Wallet PID/(Q)AEE, is available in clause 7.2.4.

6.5.4 ISO/IEC 18013-7 (unattended flow)

ISO/IEC 18013-7 [i.88] draft standard extends ISO/IEC 18013-5 [i.87] with the unattended flow, i.e. the online flow whereby an ISO mDL app connects directly to an mDL reader that is hosted as a web server application. ISO/IEC 18013-7 is backward compatible with the protocols specified in ISO/IEC 18013-5 [i.87].

ISO/IEC 18013-7 [i.88] unattended flow is based on the following protocols:

- Device Retrieval from an ISO mDL app to a web server application by using REST APIs over HTTPS POST; this flow is described in clause 7.2.5.1.
- OpenID for Verifiable Presentations (OID4VP) [i.106] in conjunction with Self-issued OpenID Provider v2 (SIOP2) [i.107]; this flow is described in clause 7.2.5.2.

Both protocols for the unattended flow transmit the selectively disclosed ISO mDL attributes in conjunction with the MSO from the ISO mDL app to the ISO mDL reader. The ISO mDL attributes and the MSO are verified according to the same principles as for the ISO mDL device retrieval flow (see clause 7.2.3).

As described in clause 6.5.1, the MSO can be signed with SOG-IS approved cryptographic algorithms and allows for QSC algorithms for future use.

ISO/IEC 18013-7 [i.88] is still a draft, so there are no real deployments in production. NIST NCCoE will carry out interoperability tests [i.103] with an ISO/IEC 18013-7 [i.88] compatible reader during the course of 2023 and 2024.

ISO/IEC 18013-7 [i.88] is not referred to by the ARF v1.0.0 [i.34], although the associated specification ISO/IEC 23220-4 [i.89] is mentioned in the ARF.

An extensive analysis of the ISO mDL unattended flow, and how it can be applied for eIDAS2 QTSPs and EUDI Wallet PID/(Q)AEE, is available in clause 7.2.5.

6.6 ISO/IEC 23220-4

ISO/IEC 23220-4 [i.89] is a draft specification describing operational (presentation) protocols for a digital wallet. The specification expands on ISO/IEC 18013-5 [i.87] with reader engagement, internet online connections to a reader, and bridges to additional standards for user authorization such as OID4VP [i.106] and credential formats such as W3C Verifiable Credentials [i.128].

ISO/IEC 23220-4 [i.89] presentation protocols are based on the following protocols:

- Device Retrieval from a digital wallet to a web server application by using REST APIs over HTTPS POST.
- OpenID for Verifiable Presentations (OID4VP) [i.106] in conjunction with Self-issued OpenID Provider v2 (SIOP2) [i.107].

More specifically, Annex B in ISO/IEC 18013-7 [i.88] draft specification refers to ISO/IEC 23220-4 [i.89] for the OID4VP/SIOP2 profile to be used for presentation of the ISO mDL and an MSO in an ISO/IEC 18013-7 [i.88] unattended flow. As described in clause 6.5.1, the MSO can be signed with SOG-IS approved cryptographic algorithms and allows for QSC algorithms for future use.

Furthermore, Annex B in ISO/IEC 23220-4 [i.89] WD9 describes how to present W3C Verifiable Credentials [i.128] in conjunction with IETF SD-JWT [i.76] for selective disclosure. The SD-JWT can be signed with SOG-IS approved cryptographic algorithms and allows for QSC algorithms for future use (see clause 7.3).

In order to secure the HTTPS connection to an online reader (relying party), ISO/IEC 23220-4 recommends the use of QWACs.

ISO/IEC 23220-4 [i.89] is still a draft, so there are no real deployments in production. However, the ARF v1.0.0 [i.34] refers to ISO/IEC 23220-4 [i.89] as an alternative attestation exchange REST API protocol.

6.7 U-Prove

The U-Prove scheme is based on ABC (Attribute Based Credentials), which in turn relies upon Stefan Brand's cryptographic research on selective disclosure and blinded signature schemes in the book "Rethinking Public Key Infrastructures and Digital Certificates; Building in Privacy" from 2000 [i.16]. Brands founded a company to implement the U-Prove ABC scheme, and this company was later acquired by Microsoft®. In 2013, Microsoft® Research released the Identity Metasystem with support for U-Prove ABC to cater for anonymous credentials [i.100]. The U-Prove ABC system was also selected by the EC-funded project Attribute based Credentials for Trust (ABC4Trust) [i.70].

In summary, the U-Prove scheme contains two protocols: Issuing the credential to a user and presenting it when accessing a relying party. The U-Prove scheme is illustrated in Figure 15.

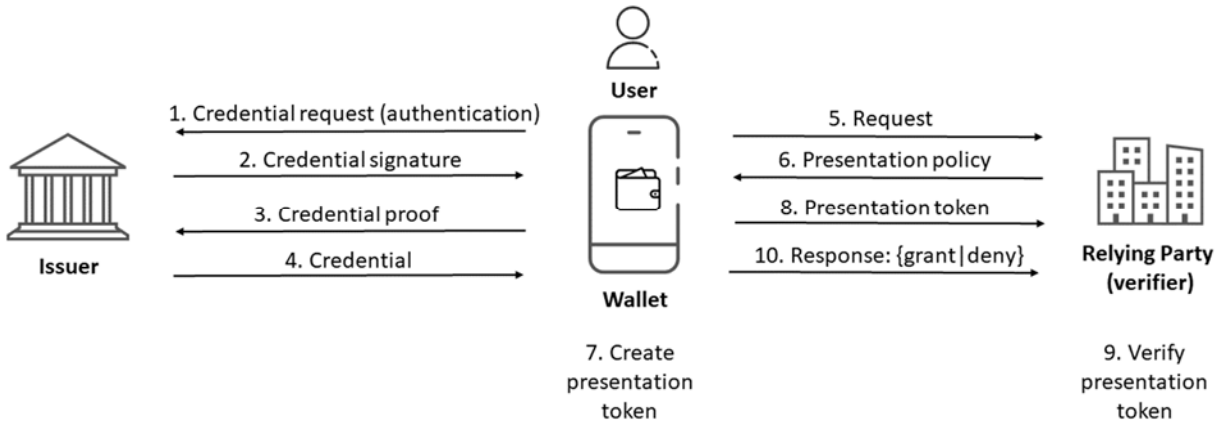

Figure 15: Overview of the U-Prove ABC scheme

The U-Prove issuing protocol is performed between the issuer and the user. The objective of this protocol is for the user to receive a credential, such that it can later present a selected set of attributes to access a relying party. The issuer basically applies a blind signature to the credential with attributes. In other words, the issuer verifies the validity of the attributes and applies a signature without seeing the resulting signature. Since the issuer does not store the result of the issuing protocol, the user cannot be tracked when using the credential, i.e. the processes of issuing and presenting are unlinkable.

The U-Prove presentation phase is based on a selective disclosure protocol between the user and the relying party. Based on the relying party's presentation policy, the user selects those attributes that it is willing to present from the issued credential. All the other attributes can be proved by the user to be unchanged in the credential. By the end of the interaction the relying party receives a presentation token with all the revealed attributes and the intact issuer's signature on the whole set of attribute values.

NOTE 1: U-Prove is similar to the Idemix (see clause 6.4) in the sense that both protocols are based on privacy-preserving ABC technology, although the iterations in the issuance phase and the underlying cryptographic algorithms differ.

The U-Prove scheme is based on the Discrete Logarithm (DL) problem and the credentials are issued as DLREP-based certificates as well as for RSAREP-certificates.

NOTE 2: Since U-Prove is based on algorithms using the Discrete Logarithm (DL) problem, the scheme cannot be considered as quantum-safe.

7 Implications of selective disclosure on standards for (Q)EAA/PID

7.1 General implications

The purpose of clause 7 is to analyse the implications of selective disclosure and unlinkability on ETSI standards for (Q)EAAs and PIDs.

More specifically, the (Q)EAA/PID credentials discussed in the following clauses 7.2 and 7.3 are scoped to ISO/IEC 18013-5 [i.87] mDL and SD-JWT, because these formats are explicitly specified as selective disclosure formats for PIDs in the ARF [i.34].

The main reason why ISO mDL and SD-JWT were selected in the ARF [i.34] as (Q)EAA/PID credentials is that they can be signed with cryptographic algorithms that are currently approved by SOG-IS [i.115], and that the credentials also allow for being signed with Quantum-Safe Cryptography (QSC) algorithms for future use. More technical details on how the issuer may apply such signatures on ISO mDL and SD-JWT are discussed in clauses 7.2.1 and 7.3.1 respectively.

The analysis is primarily focused on selective disclosure and unlinkability since those characteristics are defined in eIDAS2 [i.54] and the ARF outline [i.33]. Predicates are described on a high level, with proposals on how to implement them for the selected PID credentials ISO mDL and SD-JWT. Zero-knowledge proofs and range proofs are however out of scope for this analysis, because the selected PID credentials do not support these features.

The selected (Q)EAA/PID credentials are analysed with respect to the issuance by a QTSP/PIDP, how the credentials are stored in the EUDI Wallet, and how selected attributes are presented to a relying party.

Firstly, it is analysed how the QTSP or PID provider may issue (Q)EAAs/PIDs with capabilities for selective disclosure. This analysis also describes the PKI trust models for the issuance process and whether EU Trusted Lists (EU TLs) can be applied. Furthermore, it is described how the (Q)EAAs/PIDs should be issued to cater for unlinkability. The recommended policies and practices for such QTSP/PIDP issuance processes are discussed for ISO mDL in clause 7.2 and SD-JWT in clause 7.3.

Secondly, it is analysed how the (Q)EAAs/PIDs with capabilities for selective disclosure and unlinkability are stored in the EUDI Wallet. This analysis also describes the associated cryptographic keys used for proving the user's ownership of the (Q)EAAs/PIDs. The implications for storing the (Q)EAAs/PIDs with selective disclosure in an EUDI Wallet are discussed for ISO mDL in clause 7.2 and SD-JWT in clause 7.3.

Thirdly, it is analysed how the selected attributes can be presented to a relying party, yet sustaining unlinkability. The recommended policies and practices for presenting the (Q)EAAs/PIDs with an EUDI Wallet are discussed for ISO mDL in clause 7.2 and SD-JWT in clause 7.3.

7.2 Implications for ISO mDL with selective disclosure

7.2.1 QTSP/PIDP issuing ISO mDL

7.2.1.1 General

The ISO mDL, as specified in ISO/IEC 18013-5 [i.87], is composed by the ISO mDL data with the user's elements, the ISO mDL authentication key, and the Mobile Security Object (MSO) with a signed list of salted hash values of these elements. The MSO is a CDDL-encoded [i.85] object, which is signed by the issuer with a COSE-formatted signature [i.83].

ISO/IEC 18013-5 [i.87] describes the Issuing Authority Certification Authority (IACA) that is the root CA that used for issuing subordinated certificates, which in turn are used for signing the user's ISO mDL MSOs, signing revocation data (OCSP-responses and CRLs), and securing online services (JWS and TLS).

The clauses below compare and map the requirements on ISO mDL compliant IACAs into considerations for eIDAS2 compliant QTSPs/PIDPs when issuing ISO mDL with capabilities for selective disclosure and (predetermined) predicates. The clauses below also provide a summary of the ISO mDL and its Issuing Authorities, but it is recommended to have studied the ISO/IEC 18013-5 [i.87] before to have an understanding of the ISO mDL ecosystem.

7.2.1.2 Certificate profiles

The IACA's trust anchor is a DER-encoded X.509 certificate that should be issued according to the certificate profile in ISO/IEC 18013-5 [i.87], Annex B.1. ISO/IEC 18013-5 [i.87], Annex B.1.1 declares that all X.509 certificates are DER-encoded and specifies the generic certificate requirements on certificate extensions and subjects. The IACA certificate profile also defines the cryptographic algorithms that are approved by ISO/IEC 18013-5 [i.87].

In the context of eIDAS2, the cryptographic algorithms used in the QTSP/PIDP CA certificates are required to comply with the SOG-IS list of EU approved cryptographic algorithms [i.115]. Hence, the QTSP/PIDP CA certificates used for issuing ISO mDLs are required to comply with the intersection of IACA's and SOG-IS' requirements on cryptographic algorithms.

EXAMPLE1: SOG-IS [i.115], section "4.3 Discrete Logarithm in Elliptic Curves" lists the following approved ECC curves: BrainpoolP256r1, BrainpoolP384r1, and BrainpoolP512r1.

EXAMPLE2: ISO/IEC 18013-5 [i.87] "Table B.3 - Document signer certificate" lists the following approved ECC curves: BrainpoolP256r1, BrainpoolP320r1, BrainpoolP384r1, BrainpoolP512r1, Curve P-256, Curve P-384, and Curve P-521.

The IACA trust anchor is used for issuing the following subordinated certificates in an IACA PKI:

- mDL MSO signer certificate (ISO/IEC 18013-5 [i.87], Annex B1.2).
- JWS signing certificate (ISO/IEC 18013-5 [i.87], Annex B.1.3.1).
- TLS server certificate issuing authority (ISO/IEC 18013-5 [i.87], Annex B1.6).
- TLS client authentication certificate (ISO/IEC 18013-5 [i.87], Annex B.1.8).
- OCSP signer certificate (ISO/IEC 18013-5 [i.87], Annex B.1.9).

Furthermore, the ISO mDL IACA CRL profile is specified in Annex B.2 in ISO/IEC 18013-5 [i.87].

An eIDAS2 QTSP/PIDP that issues ISO mDLs should adhere to the IACA PKI and the certificate and CRL profiles described above.

One more alternative could be for ETSI to assign a specific QC extension to be used for trust anchor certificates that are used by accredited QTSPs to issue ISO mDLs.

7.2.1.3 Trusted Lists

According to article 22(1) of eIDAS [i.55], each EU Member State is required to publish a Trusted List (TL) with all QTSPs in that EU Member State. All information referred to in eIDAS article 22(3), including the location and signing certificates of the TLs, is compiled in the EU LOTL (List Of Trusted Lists). Furthermore, the Commission Implementing Directive (CID) 2015/1505 [i.52] mandates the use of ETSI TS 119 612 [i.47] for the implementation of the trusted lists. ETSI TS 119 612 [i.47] specifies the format and mechanisms for establishing, locating, accessing and authenticating trusted lists. The EU TLs and EU TOTL are XML-encoded according to specific XML schemas and signed with XAdES-signatures as specified in ETSI TS 119 612 [i.47].

ISO/IEC 18013-5 [i.87] has introduced a similar concept called Verified Issuer Certificate Authority List (VICAL), which contains the trustworthy IACA's that issue certificates for creating and operating ISO mDLs. An ISO mDL VICAL can be formatted and signed either in CDDL [i.85] or CMS [i.77] format. The ISO mDL VICAL Providers publishes the VICALs. ISO/IEC 18013-5 [i.87] Annex C specifies the policy and security requirements and technical and procedural controls for a VICAL Provider.

NOTE: ISO/IEC 18013-5 [i.87], Annex C refers to ETSI EN 319 411-1 [i.44] and FPKIPA X.509 Certificate Policy For The U.S. Federal PKI Common Policy Framework [i.58] for the operations of an ISO mDL VICAL Provider.

Hence, there are synergies between the EU TLs and the ISO mDL VICALs, in the sense that both trusted lists contain trust anchors. The main differences are the encodings and signature formats (EU TL XML/XAdES versus ISO mDL VICAL CDDL/CMS). In order to bridge this gap, ETSI TS 119 612 [i.47] may specify a CDDL/CMS profile of the EU TL that is compatible with the ISO mDL VICAL, or ISO/IEC 18013-5 [i.87] may be extended to specify an XML profile of the VICAL that is compatible with the ETSI EU TLs. In such a scenario, an eIDAS2 accredited QTSP/PIDP could issue CA certificates that are included in an EU TL, which in turn could be trusted as a VICAL in the ISO mDL ecosystem.

In summary, transposing ISO/IEC 18013-5 [i.87], Annex C to an eIDAS2 context results in the following recommendations:

- The ISO mDL Issuing Authority corresponds to the eIDAS2 QTSP/PIDP.
- The IACA trust anchor should be issued as a trust anchor by the eIDAS2 QTSP/PIDP that issues ISO mDL as (Q)EAA/PID.
- The eIDAS2 QTSP/PIDP should ensure that its IACA trust anchor is published in the EU TL, which is issued by the supervisory body in the applicable EU Member State.
- ETSI TS 119 612 [i.47] may specify an additional CDDL/CMS profile of the EU TL that is compatible with the ISO mDL VICAL, or ISO/IEC 18013-5 [i.87] may be extended to specify an XML profile of the VICAL that is compatible with the ETSI EU TLs.
- The EU TLs may include a specific extension for the QTSPs that are authorized to issue QEAAs that also are compliant with ISO mDL; the EU TL extension can reference the ISO mDL VICAL where the QTSP is also listed.

7.2.1.4 Issuance of ISO mDLs

An ISO mDL, which has been issued to the user's EUDI Wallet on a device, is essentially composed of the mDL data and the MSO, which are associated with the mDL authentication key (see clause 7.2.2).

The ISO mDL data is an unsigned list of the user's elements belonging to the nameSpace "org.iso.18013.5.1", as defined in ISO/IEC 18013-5 [i.87].

The MSO (mobile security object) is defined in ISO/IEC 18013-5 [i.87], section 9.1.2.4 as a signed object, which contains the mDL authentication public key and a list of salted hash values of the user's elements. The MSO is signed with a COSE-formatted signature, by the IACA's MSO signer certificate.

NOTE 1: In the context of eIDAS2, a QTSP/PIDP will issue an MSO signer certificate with cryptographic algorithms that are approved by both SOG-IS [i.115] and the ISO/IEC 18013-5 [i.87].

NOTE 2: Since the MSO's signature is COSE-formatted, QSC algorithms such as CRYSTALS Dilithium, FALCON, and SPHINCS+ can also be considered for the future according to the IETF IESG report [i.73].

In order to achieve unlinkability when presenting the ISO mDL elements multiple times, it is required that the mDL authentication keys and related MSOs are updated frequently (see ISO/IEC 18013-5 [i.87], section E.8.4). Hence, the QTSP/PIDP should establish processes for issuing multiple MSOs to the user's EUDI Wallet, typically in batches prior to the device retrieval use of the MSOs. The EUDI Wallet may also signal to the QTSP/PIDP when it is necessary to refresh the MSOs. When issuing a new MSO, the random salts in IssuerSignedItems for the hash calculations should be unique such that the random salted hash values differ for each MSO, even if the user's ISO mDL data elements remain the same.

EXAMPLE 1: Assume that the user's GivenName in the ISO mDL data is "Smith". If the GivenName is combined with random salt S1 and hashed, the resulting hash value becomes H1 in the first MSO. If the same GivenName name is combined with another random salt S2 and hashed, the resulting hash value becomes H2 in the second MSO.

ISO mDL does not support predicates in the sense that Zero-Knowledge Proofs or range proofs can be dynamically derived based on the elements in the ISO mDL data. However, ISO/IEC 18013-5 [i.87], clause 7.2.5 specifies the possibility to insert predetermined boolean elements as "age_over_NN" in the ISO mDL.

EXAMPLE 2: The boolean statement "age_over_18" could be an element in the ISO mDL data.

NOTE 3: It is possible to include signed computational inputs and parameters to enable dynamic predicates (see clause 4.5.2.4).

In order to achieve (predetermined) predicates, the issuing QTSP/PIDP should establish processes to identify the relevant boolean statements and insert them as elements in the ISO mDL.

7.2.1.5 Comparison with ETSI certificate profiles for Open Banking (PSD2)

ETSI ESI has specified certificate profiles and TSP policy requirements for Open Banking in the sector specific ETSI TS 119 495 [i.46]. The scope of ETSI TS 119 495 [i.46] is:

- Specifies requirements for qualified certificates for electronic seals and website authentication, to be used by payment service providers in order to meet needs of Open Banking including the EU PSD2 [i.53] Regulatory Technical Standards (RTS) [i.51].
- Specifies additional TSP policy requirements for the management (including verification and revocation) of additional certificate attributes as required by the above profiles.

In summary, a QTSP can issue PSD2 compliant certificates (QWACs or QCert for eSeal), using the certificate profile specified in ETSI TS 119 495 [i.46] as follows. The PSD2 specific attributes are checked by the (Q)TSP as part of the identity proofing, as specified in the ETSI TS 119 495 [i.46] REG-6.2.2-1, which states: *"The TSP shall verify the Open Banking Attributes (see clauses 5.1 and 5.2) provided by the subject using authentic information from the Competent Authority (e.g. a national public register, EBA PSD2 Register, EBA Credit Institution Register, authenticated letter)."* The EBA (European Banking Association) maintains a register of payment institutions [i.41], which can be used for that purpose. As a result, a QCStatement extension with Open Banking attributes is included in the PSD2 certificate, which proves its compliance with the PSD2 RTS.

A relying party intending to validate a PSD2 certificate usually performs a two step validation approach:

1) The relying party validates the qualified status of the certificate using the EU TLs.
2) The relying party confirms the correctness of the PSD2 attributes included in the certificate QCStatement using either the national public registers, or the EBA register. The relying parties need to have out-of-band knowledge of where to retrieve the EBA register.

The ETSI TS 119 495 [i.46] requirements for (Q)TSPs issuing PSD2 certificates may partially be re-used also for the issuance of ISO mDLs, but with the following differences:

- The format will be (Q)EAA for ISO mDL instead of X.509 certificates.

- The relying party will confirm that the QTSP having issued the (Q)EAA is authorized to issue this specific type of (Q)EAA by looking into a domain-specific list, i.e. the ISO mDL VICAL.
- To facilitate the validation of (Q)EAAs being used ISO mDLs, EU TLs could be used to point towards the domain-specific VICAL list where a QTSP is listed as being authorized for a specific scope. Alternatively, an URI for accessing this domain-specific VICAL list could be included in the ISO mDL (Q)EAA itself, although this may be too static as this URI may change over time.

7.2.1.6 Mapping of ISO mDL and eIDAS2 terms

As discussed in the clauses above, there are several equivalences between the terms in ISO/IEC 18013-5 [i.87] and the terms in eIDAS2 [i.54] and the ARF [i.34].

Table 2 provides a mapping of eIDAS2 and ARF terms with the syntax used in ISO/IEC 18013-5 [i.87].

Table 2: Mapping of eIDAS2/ARF and ISO/IEC 18013-5 [i.87] terms

Terms in eIDAS2 and the ARF	Terms in ISO/IEC 18013-5 (mDL)
End users of EUDI Wallets	mDL Holder
EUDI Wallet issuers	Technology Providers
Person Identification Data Providers	Issuing Authorities
Providers of registries of trusted sources (e.g. EU TL)	Verified Issuer Certificate Authority List (VICAL) Providers
Qualified and non-qualified electronic attestation of attributes (qEAA) providers	Issuing Authorities
QTSPs for issuing qualified and non-qualified certificate for electronic signature/seal providers	Issuing Authority Certification Authority (IACA)
Providers of other trust services	Not defined
Authentic sources	Governmental authoritative source
Relying parties	mDL Reader, operated by a mDL verifier
Conformity Assessment Bodies (CAB)	Auditing Bodies following ISO/IEC 27001 [i.91] and ISO/IEC 27002 [i.92]
Supervisory bodies	Auditing Bodies following ISO/IEC 27001 [i.91] and ISO/IEC 27002 [i.92]
Device manufacturers and related subsystems providers	Technology Providers
Catalogue of attributes and schemes for the attestations of attribute providers	ISO mDL namespace

7.2.2 EUDI Wallet mDL authentication key

The mDL authentication key is used to prevent cloning of the ISO mDL and to mitigate man in the middle attacks. The mDL authentication key pair consists of a public and a private key denoted as (SDeviceKey.Priv, SDeviceKey.Pub). The mDL authentication public key is stored as the DeviceKey element in the MSO, and the corresponding mDL authentication private key is used for signing the response data contained in the DeviceSignedItems structure (see ISO/IEC 18013-5 [i.87], clauses 9.1.3, 9.1.2.4 and 9.1.3.3 for more information).

Hence, the mDL authentication key is used by the EUDI Wallet for authentication of selectively disclosed mDL data elements that are presented to a relying party (see clause 7.2.3).

More information on how to store the ISO mDL data, MSO, and the mDL authentication key is available in clause 7.4.

7.2.3 EUDI Wallet used with ISO mDL device retrieval flow

7.2.3.1 Overview of the ISO mDL device retrieval flow

The scope of the present clause is to describe how the EUDI Wallet can present ISO mDL selectively disclosed elements over the ISO mDL device retrieval flow, and how eIDAS2 trust services can be used to support this process.

NOTE: The ISO mDL device retrieval flow is mandatory for the EUDI Wallet according to the ARF [i.34].

The ISO mDL device retrieval flow is described in ISO/IEC 18013-5 [i.87], clauses 6.3.2, 6.3.2.1 (as flow 1) and 6.3.2.4. This clause will not repeat the entire ISO mDL device retrieval process, although a brief summary is provided below for readability with references to the ISO/IEC 18013-5 [i.87].

The ISO mDL device retrieval flow is illustrated in Figure 16.

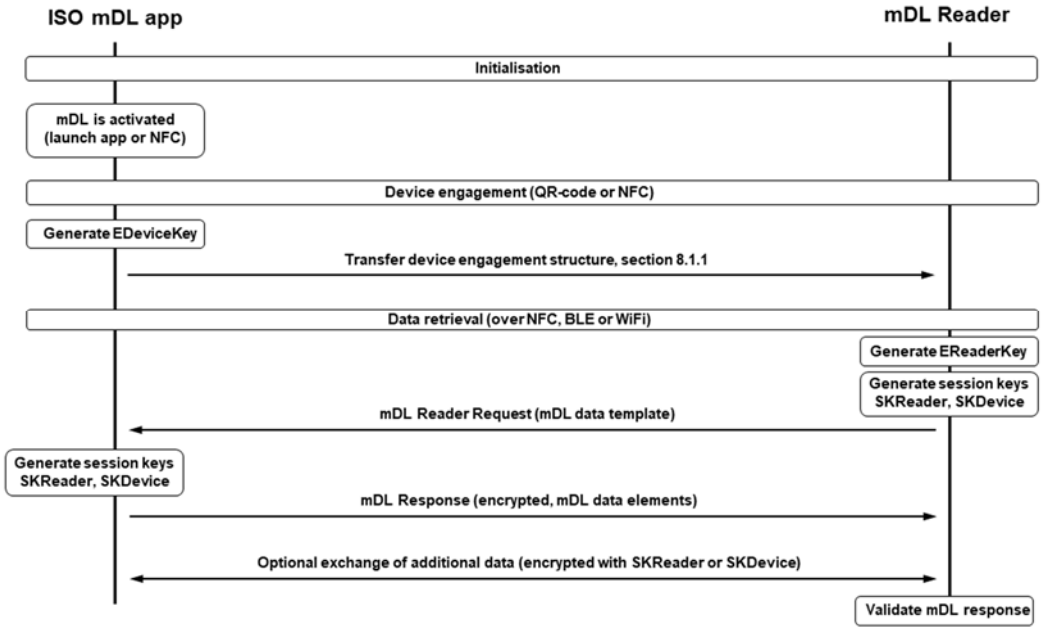

Figure 16: Overview of the ISO mDL device retrieval flow

On a high level, the ISO mDL device retrieval flow can be divided in the following phases, where the ISO mDL reader is equivalent to an attended eIDAS2 relying party:

- Initialization phase, whereby the ISO mDL app is activated either by the user or triggered by NFC contact with the ISO mDL reader (see ISO/IEC 18013-5 [i.87], clause 6.3.2.2 for more information).
- Device engagement phase, whereby the ephemeral device key EDeviceKey is generated, and the device engagement structure is transferred over NFC or as QR-code. The device engagement structure contains parameters for device retrieval transfer options TransferMethod and TransferOptions (see ISO/IEC 18013-5 [i.87], clauses 6.3.2.3, 9.1.1, 8.2.1, 8.2.2 and 8.2.1.1 for more information).
- Data retrieval phase, whereby the EReaderKey, SKReader and SKDevice keys are generated to establish an encryption session. The ISO mDL reader then transmits the mDL Reader Request and the ISO mDL replies with the mDL Response (see ISO/IEC 18013-5 [i.87], clauses 9.1, 9.1.1, 8.3.2.1.2 and 8.3.2.2.2 for more information).

As regards to selective disclosure, the mDL Reader Request contains a list of the DataElements the mDL Reader requests from the mDL app. Upon the user's consent, the mDL app will reply with the mDL Response with the selected DataElements in the DeviceSignedItems. The DeviceSignedItems object is signed by the mDL Authentication Key, to which the user is authenticated with a PIN-code or biometrics (see ISO/IEC 18013-5 [i.87], clauses 8.3.2.1.2 and 8.3.2.2.2 for more information).

The selected DataElements will be hashed at the mDL reader, and be compared with the corresponding hash values in the MSO. ISO/IEC 18013-5 [i.87], clause 9.1.2.3 describes how the relying party validates the MSO signature and how to check that the hashed mDL data elements match the hash values in the MSO.

More specifically, ISO/IEC 18013-5 [i.87], clause 9.1.2.3 specifies in detail how the mDL reader validates the certificate chain of the IACA trust anchor and the Issuing Authority's MSO signer certificate. ISO/IEC 18013-5 [i.87], Annex C describes the ISO mDL VICAL, which points to the IACA trust anchor and revocation information.

7.2.3.2 Analysis of the ISO mDL device retrieval flow applied to eIDAS2

An analysis of the ISO mDL device retrieval flow applied to an eIDAS2 context results in the following observations and recommendations:

- The ISO mDL app should be part of an EUDI Wallet.
- The ISO mDL Issuing Authority corresponds to a QTSP, PIDP and/or an EUDI Wallet provider.
- The mDL Reader corresponds to an device retrieval eIDAS2 relying party (that will validate the ISO mDL as an (Q)EAA/PID).
- The recommendations should be observed in clause 7.2.1 on how a QTSP/PIDP supervised under eIDAS2 can operate as an ISO mDL IACA.
- The recommendations should be observed in clause 7.2.1 on how an eIDAS2 EU TL should be formatted to be compatible as an ISO mDL VICAL or vice versa.
- The eIDAS2 relying party should use the eIDAS2 EU TL (which is equivalent to an ISO mDL VICAL) to retrieve the QTSP/PIDP trust anchor (which is equivalent to the IACA trust anchor).
- The eIDAS2 relying party should validate the MSO (submitted by the ISO mDL app in the mDL Response) according to the principles in ISO/IEC 18013-5 [i.87], clause 9.1.2.3, by using the QTSP/PIDP trust anchor.
- The MSOs in the EUDI Wallet ISO mDL app should be unique as described in clause 7.2.1 to cater for unlinkability when validated by the relying party.
- The MSO is signed by the QTSP/PIDP with a COSE formatted signature, which allows for SOG-IS approved cryptographic algorithms [i.115] and for QSC for future use [i.73].

These observations and recommendations should be considered with respect to selective disclosure for the ETSI work items ETSI TS 119 462 [i.48], ETSI TS 119 471 [i.49] and ETSI TS 119 472 [i.50].

7.2.4 EUDI Wallet used with ISO mDL server retrieval flow

7.2.4.1 Overview of the ISO mDL server retrieval flows

The scope of the present clause is to describe how the EUDI Wallet can present ISO mDL selectively disclosed elements over the ISO mDL server retrieval flow, and how eIDAS2 trust services can be used to support this process.

NOTE: This ISO mDL server retrieval flow is NOT mentioned by the ARF, but may need to be used by national or specific implementations that need to be interoperable with ISO mDL.

The ISO mDL server retrieval flow can be initialized as a hybrid device/server process (see clause 7.2.4.2) or as a server process (see clause 7.2.4.3). Once the ISO mDL server retrieval flow has been initialized, it continues with either the WebAPI (see clause 7.2.4.5) or the OpenID Connect (OIDC) flow (see clause 7.2.4.7). Clause 7.2.4 will not repeat the entire ISO mDL server retrieval process, although a brief summary is provided below for readability with references to the ISO/IEC 18013-5 standard.

7.2.4.2 ISO mDL flow initialization

The initialization of the ISO mDL device and server retrieval flows are described in ISO/IEC 18013-5 [i.87], clauses 6.3.2, 6.3.2.1 (as flow 2) and 6.3.2.4.

The ISO mDL device/server data retrieval flow is illustrated in Figure 17.

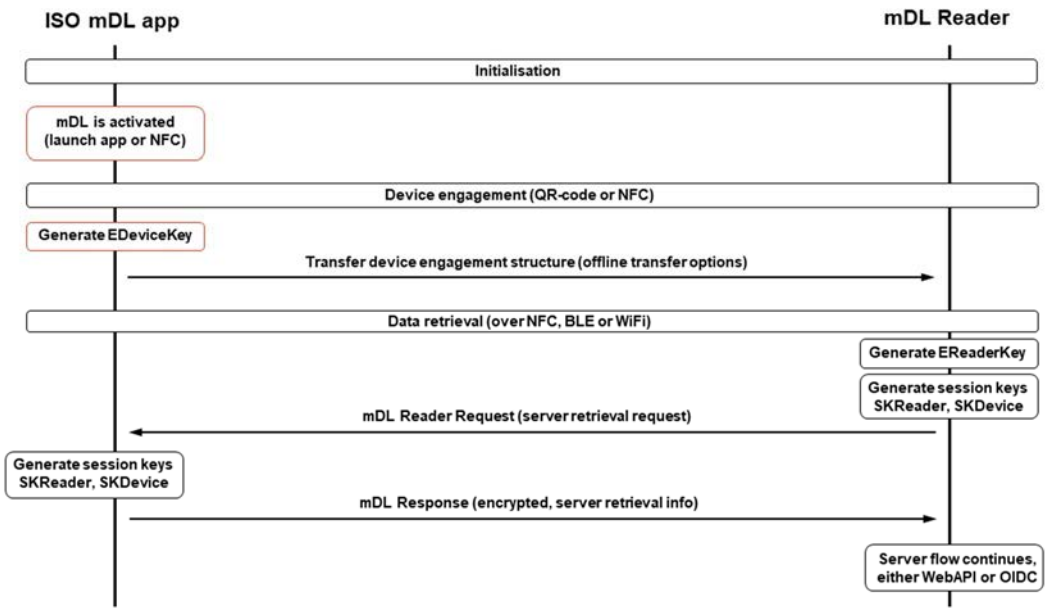

Figure 17: ISO mDL flow initialization

On a high level, the ISO mDL device/server retrieval flow can be divided in the following phases (where the ISO mDL reader is equivalent to an eIDAS2 relying party):

- Initialization phase, whereby the ISO mDL app is activated either by the user or triggered by NFC contact with the ISO mDL reader (see ISO/IEC 18013-5 [i.87], clause 6.3.2.2 for more information).
- Device engagement phase, whereby the ephemeral device key EDeviceKey is generated, and the device engagement structure is transferred over NFC or as QR-code (see ISO/IEC 18013-5 [i.87], clauses 6.3.2.3, 9.1.1, 8.2.1 and 8.2.2 for more information).
- Data retrieval phase, whereby the EReaderKey, SKReader and SKDevice keys are generated to establish an encryption session. The ISO mDL reader then transmits the mDL Reader Request including the server retrieval request and the ISO mDL replies with the mDL Response including the server retrieval information (see ISO/IEC 18013-5 [i.87], clauses 9.1, 9.1.1, 8.3.2.1.2.1 and 8.3.2.1.2.2 for more information).

The ISO mDL online data retrieval flow continues with either the WebAPI (see clause 7.2.4.5) or OIDC (see clause 7.2.4.7).

7.2.4.3 ISO mDL server retrieval flow initialization

The ISO mDL server retrieval flow initialization is described in ISO/IEC 18013-5 [i.87], clauses 6.3.2, 6.3.2.1 (as flow 3) and 6.3.2.4.

The ISO mDL server retrieval flow initialization is illustrated in Figure 18.

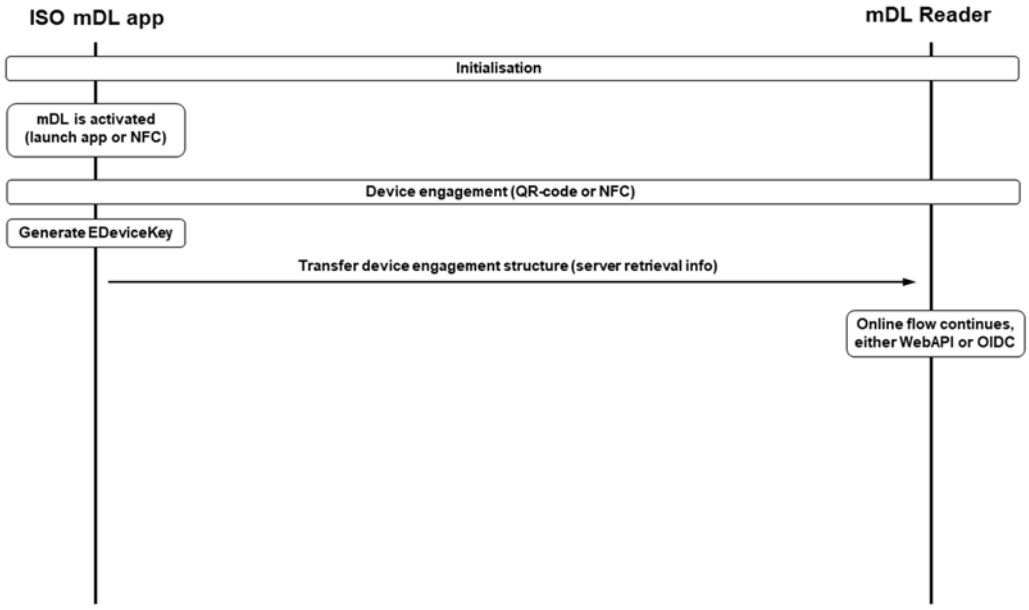

Figure 18: ISO mDL server retrieval flow initialization

On a high level, the ISO mDL server retrieval flow can be divided in the following phases (where the ISO mDL reader is equivalent to an eIDAS2 relying party):

- Initialization phase, whereby the ISO mDL app is activated either by the user or triggered by NFC contact with the ISO mDL reader (see ISO/IEC 18013-5 [i.87], clause 6.3.2.2 for more information).
- Device engagement phase, whereby the ephemeral device key EDeviceKey is generated, and the device engagement structure is transferred over NFC or as QR-code. The device engagement structure contains parameters for online transfer options WebAPI or OIDC (see ISO/IEC 18013-5 [i.87], clauses 6.3.2.3, 9.1.1, 8.2.1, 8.2.2 and 8.2.1.1 for more information).

The ISO mDL server retrieval flow continues with either the WebAPI (see clause 7.2.4.5) or OIDC (see clause 7.2.4.7).

7.2.4.4 ISO mDL server retrieval WebAPI flow

The ISO mDL server retrieval flow is described in ISO/IEC 18013-5 [i.87], clause 8.3.2.2 and the WebAPI calls are specified in ISO/IEC 18013-5 [i.87], clause 8.3.2.2.2.

The ISO mDL WebAPI server retrieval flow is illustrated in Figure 19.

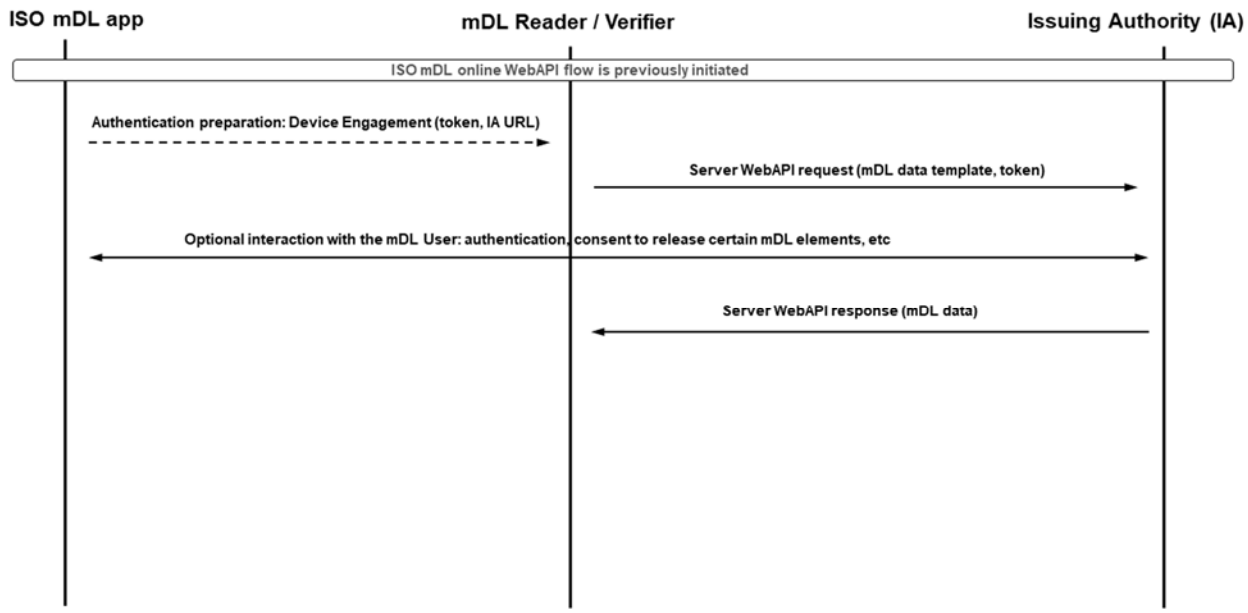

Figure 19: ISO mDL server retrieval WebAPI flow

As regards to selective disclosure, the mDL Reader submits a server retrieval WebAPI Request with a list of requested DataElements to the Issuing Authority. Upon the user's consent, the Issuing Authority will reply with the mDL Response with the selected and disclosed DataElements (see ISO/IEC 18013-5 [i.87], clause 8.3.2.2.2 for more information).

7.2.4.5 Analysis of the ISO mDL server retrieval WebAPI flow applied to eIDAS2

An analysis of the ISO mDL WebAPI server retrieval flow applied to an eIDAS2 context results in the following observations and recommendations:

- The ISO mDL app should be part of an EUDI Wallet.
- The ISO mDL Issuing Authority corresponds to a QTSP, PIDP and/or an EUDI Wallet provider.
- The mDL Reader corresponds to an eIDAS2 relying party, which will connect to the ISO mDL Issuing Authority over the WebAPI to request information about the user.

NOTE: eIDAS2 [i.54] Article 6a.7 states: "*The issuer of the European Digital Identity Wallet shall not collect information about the use of the wallet which are not necessary for the provision of the wallet services...*" If the ISO mDL Issuing Authority also has the role as an eIDAS2 European Digital Identity Wallet provider, the statement in eIDAS2 article 6a.7 should be considered to respect the user's privacy.

- The ISO mDL Issuing Authority may deploy QWACs in order to prove its authenticity over TLS to the connecting relying parties.
- The WebAPI token is a JWT that is signed by the ISO mDL Issuing Authority OIDC Authorization Server. The JWT signer certificate should be issued by an IACA, which in the eIDAS2 context is also a QTSP.
- The ISO mDL Reader, which is an eIDAS2 relying party, should use the ISO mDL VICAL (EU TL) to retrieve the IACA trust anchor (QTSP trust anchor).
- The WebAPI JWT is signed by the QTSP/PIDP with a JOSE formatted signature, which allows for SOG-IS approved cryptographic algorithms [i.115] and for QSC for future use [i.73].

These observations and recommendations should be considered with respect to selective disclosure for ETSI TS 119 462 [i.48], ETSI TS 119 471 [i.49] and ETSI TS 119 472 [i.50].

7.2.4.6 ISO mDL server retrieval OIDC flow

The ISO mDL server retrieval flow is described in ISO/IEC 18013-5 [i.87], clause 8.3.2.2 and the OIDC calls are specified in ISO/IEC 18013-5 [i.87], clause 8.3.3.2.2.

The ISO mDL OIDC server retrieval flow is illustrated in Figure 20.

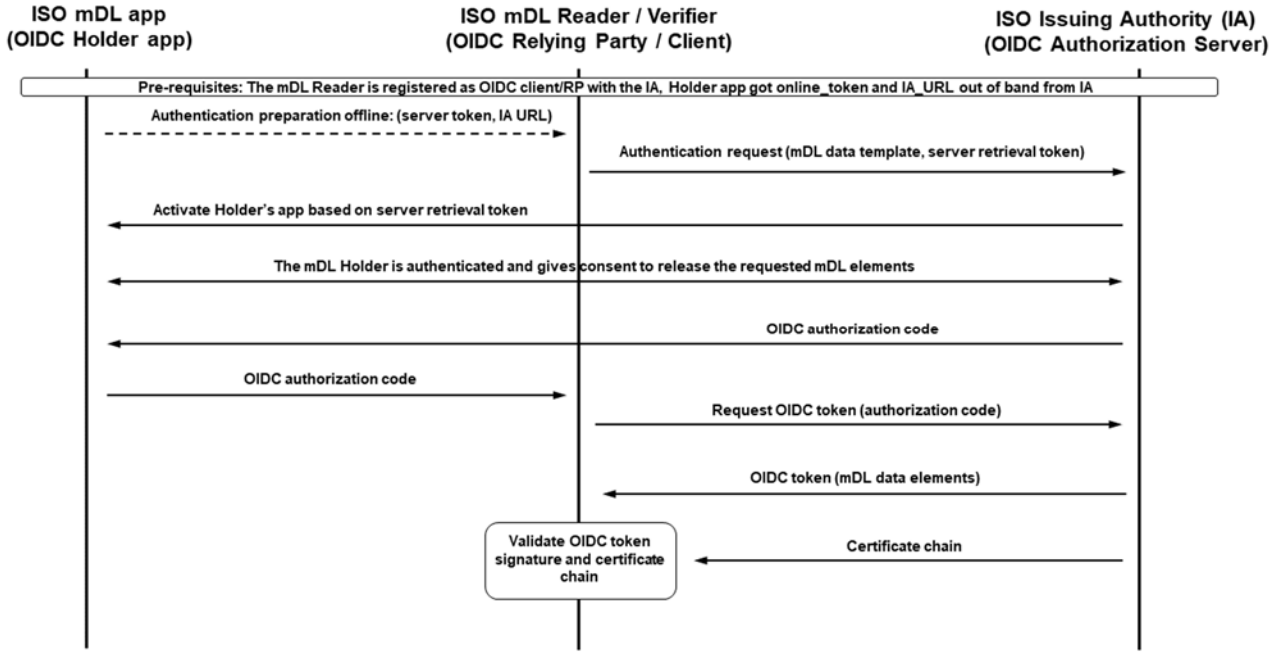

Figure 20: ISO mDL server retrieval OIDC flow

As regards to selective disclosure, the mDL Reader (OIDC client) submits an server retrieval OIDC Request with the requested data elements (JWT claims) to the Issuing Authority, which operates an OIDC Authorization Server. This activates the OIDC authorization code flow [i.105]. Based on the user's consent, the Issuing Authority (OIDC Authorization Server) will reply to the mDL Reader (OIDC client) with the OIDC Token with the selected and disclosed JWT claims about the user (see ISO/IEC 18013-5 [i.87], clause 8.3.3.2.2 and Annex D.4.2.2 for more information about the OIDC workflow).

7.2.4.7 Analysis of the ISO mDL OIDC server retrieval flow applied to eIDAS2

An analysis of the ISO mDL OIDC server retrieval flow applied to an eIDAS2 context results in the following observations and recommendations:

- The ISO mDL app should be part of an EUDI Wallet.
- The ISO mDL Issuing Authority corresponds to a QTSP, PIDP and/or an EUDI Wallet provider.
- The ISO mDL Issuing Authority operates an OIDC Authorization Server, which supports the OIDC authorization code flow.
- The mDL Reader corresponds to an eIDAS2 relying party, which is registered as an OIDC client to the ISO mDL Issuing Authority OIDC Authorization Server. The mDL Reader will connect to the ISO mDL Issuing Authority over OIDC to request information about the user.

NOTE: eIDAS2 [i.54] Article 6a.7 states: "*The issuer of the European Digital Identity Wallet shall not collect information about the use of the wallet which are not necessary for the provision of the wallet services...*" If the ISO mDL Issuing Authority also has the role as an eIDAS2 European Digital Identity Wallet provider, the statement in eIDAS2 article 6a.7 should be considered to respect the user's privacy.

- The ISO mDL Issuing Authority may deploy QWACs in order to prove its authenticity over TLS to the connecting relying parties.

- The OIDC Token is a JWT that is signed by the ISO mDL Issuing Authority OIDC Authorization Server. The JWT signer certificate should be issued by an IACA, which in the eIDAS2 context is also a QTSP.
- The ISO mDL Reader, which is an eIDAS2 relying party, should use the ISO mDL VICAL (EU TL) to retrieve the IACA trust anchor (QTSP trust anchor).
- The OIDC token JWT is signed by the QTSP/PIDP with a JOSE formatted signature, which allows for SOG-IS approved cryptographic algorithms [i.115] and for QSC for future use [i.73].

These observations and recommendations should be considered with respect to selective disclosure for ETSI TS 119 462 [i.48], ETSI TS 119 471 [i.49] and ETSI TS 119 472 [i.50].

7.2.5 EUDI Wallets used with ISO/IEC 18013-7 for unattended flow

7.2.5.1 Overview of the ISO/IEC 18013-7 flows

ISO/IEC 18013-7 [i.88] draft standard extends ISO/IEC 18013-5 [i.87] with the unattended flow, i.e. the server retrieval flow whereby an ISO mDL app connects directly to an mDL reader that is hosted as a web server application. ISO/IEC 18013-7 [i.88] is backward compatible with the protocols in ISO/IEC 18013-5 [i.87].

NOTE: Since the ISO mDL app connects directly to the web hosted mDL reader without involving any issuer, this flow preserves the user's privacy as required in eIDAS2 [i.54], Article 6a.7.

ISO/IEC 18013-7 [i.88] unattended flow is designed based on the following protocols:

- Device Retrieval from an ISO mDL app to a web server application over HTTPS POST; this flow is described in clause 7.2.5.2.
- OpenID for Verifiable Presentations (OID4VP) [i.106] in conjunction with Self-issued OpenID Provider v2 (SIOP2) [i.107]; this flow is described in clause 7.2.5.3.

7.2.5.2 ISO/IEC 18013-7 Device Retrieval flow

The general data retrieval architecture is described in ISO/IEC 18013-5 [i.87], clause 6.3.2.4. ISO/IEC 18013-7 [i.88] draft standard describes device retrieval of data for unattended (i.e. online web application) use cases. The ISO mDL app and the ISO mDL reader support device retrieval using the mDL request and response as specified in ISO/IEC 18013-5 [i.87], clause 8.3.2.1.

ISO/IEC 18013-7 [i.88] adds Annex A that specifies the Reader Engagement phase, which takes place before the Device Engagement phase in ISO/IEC 18013-5 [i.87]. The Reader Engagement struct contains the parameter RetrievalOptions, which in turn includes the RestApiOptions that defines the URI and REST API parameters for the HTTPS connection to the web hosted mDL Reader.

ISO/IEC 18013-7 [i.88] unattended online retrieval flow is illustrated in Figure 21.

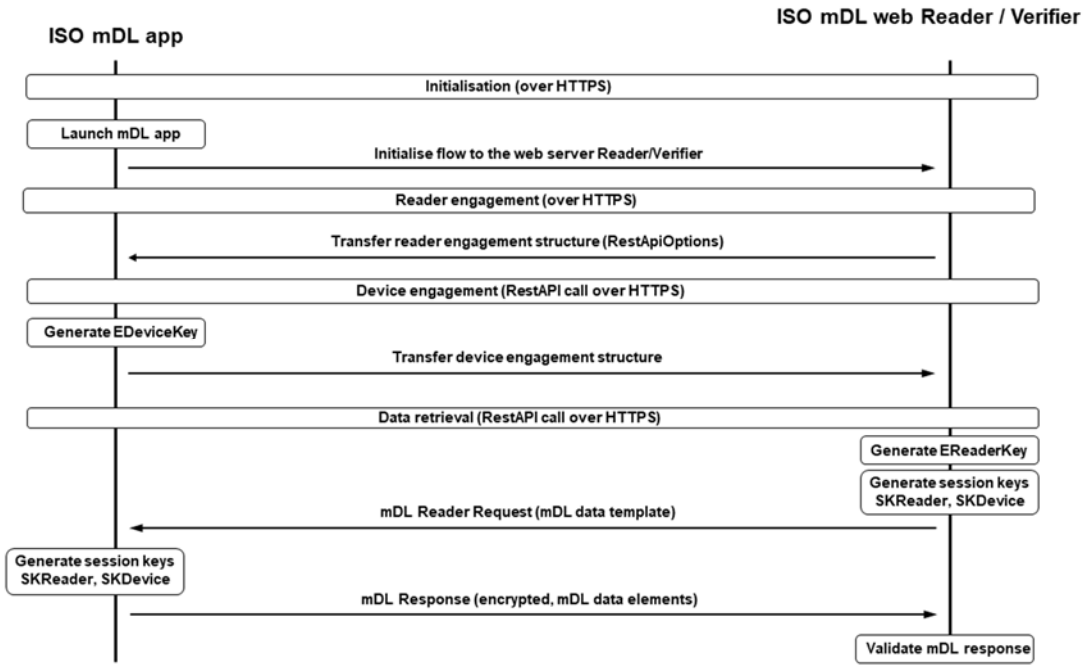

Figure 21: ISO mDL unattended Device Retrieval flow

When the mDL Response has been retrieved and parsed by the ISO mDL reader/verifier, the mDL selected attributes and MSO are verified according to the same process as the ISO mDL device retrieval flow (clause 7.2.3).

As regards to selective disclosure for the ISO mDL unattended Device Retrieval flow, the same principles and recommendations apply as for the ISO mDL device retrieval flow (clause 7.2.3). However, the ISO/IEC 18013-7 specification is not referred to by the ARF v1.0.0 [i.34], although the associated specification ISO/IEC 23220-4 is mentioned in the ARF.

7.2.5.3 ISO/IEC 18013-7 OID4VP/SIOP2 flow

As an alternative to the unattended Device Retrieval flow, ISO/IEC 18013-7 [i.88] specifies an unattended (online) flow based on OID4VP [i.106] with SIOP2 [i.107]. The OID4VP/SIOP2 flow is defined in Annex B of ISO/IEC 18013-7 [i.88]. Furthermore, the OID4VP/SIOP2 protocol is based on the ISO/IEC 23220-4 [i.89] profile for presentations of ISO mDL.

ISO/IEC 18013-7 [i.88] unattended OID4VP/SIOP2 flow is illustrated in Figure 22.

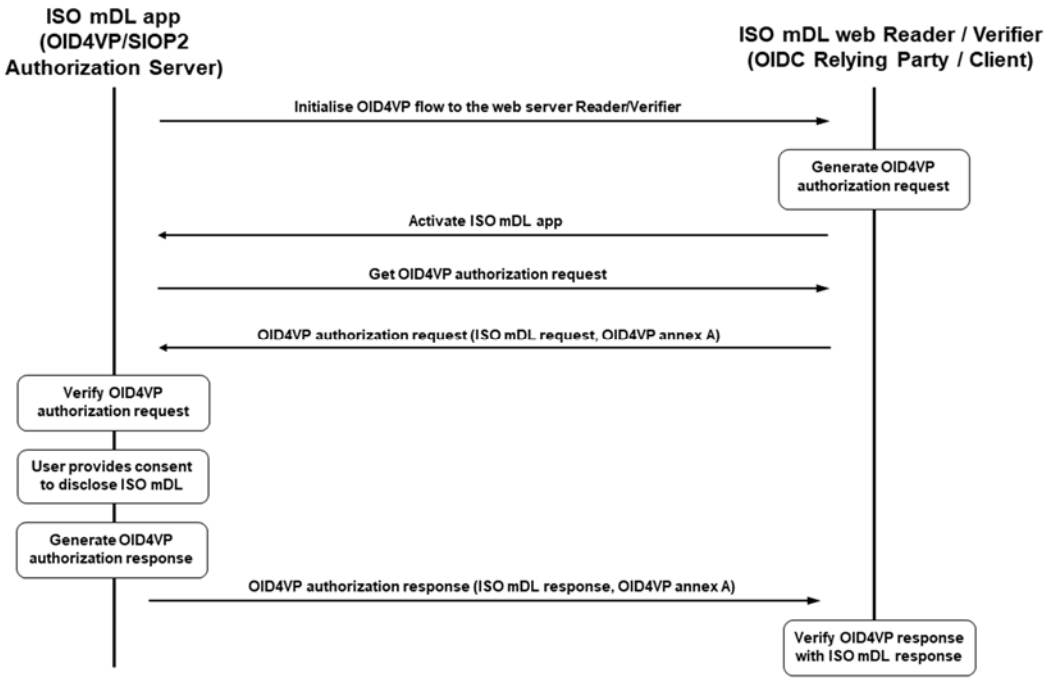

Figure 22: ISO mDL unattended OID4VP/SIOP2 flow

When the OID4VP Response, which contains the mDL Response, has been retrieved and parsed by the ISO mDL reader/verifier, the mDL selected attributes and MSO are verified according to the same process as the ISO mDL device retrieval flow (clause 7.2.3).

As regards to selective disclosure for the ISO mDL unattended OID4VP/SIOP2 flow, the same principles and recommendations apply as for the ISO mDL device retrieval flow (clause 7.2.3). However, the ISO/IEC 18013-7 [i.88] specification is not referred to by the ARF v1.0.0 [i.34], although the associated specification ISO/IEC 23220-4 [i.89] is mentioned in the ARF.

NOTE: ISO/IEC 23220-4 [i.89] is mentioned as a target in the ARF [i.34], but not mandatory since not yet published. If ISO/IEC 23220-4 [i.89] will include ISO/IEC 18013-5 [i.87] proximity as well as OID4VCI and OID4VP then 23220-4 is likely to be mandatory in a future version of the ARF.

7.3 Implications for SD-JWT selective disclosure

7.3.1 Background to W3C VCDM and SD-JWT

The ARF 1.0 text mandates the joint utilization of W3C VCDM v1.1 and SD-JWT. The former is used to express the data model and provide the overall structure of the attestation, whereas the latter is proposed as a selective disclosure capable proof mechanism.

NOTE 1: The SD-JWT specification works as a standalone attestation format too as it was not designed to provide selective disclosure capability for W3C VCs specifically but for JWTs in general.

NOTE 2: At the time of writing the present document, there is a proposal to remove the mandatory status of W3C VCDM v1.1 from the Type 1 configuration in the ARF in favor of other options. Discussing all these proposals is outside the scope of the present document, so herein the focus is only on how the proposals impact selective disclosure.

To understand the implications of SD-JWT for selective disclosure, especially in relation to the W3C VCDM, it is important to first understand what W3C VCDM is and how different proof mechanisms relate to it. After providing such a primer, the reader will be better able to understand the motivation behind the recommendations and the specific ways SD-JWT is presented herein.

7.3.2 A primer on W3C VCDM

7.3.2.1 Overview of W3C Verifiable Credential Data Model (VCDM)

The W3C Verifiable Credential Data Model (VCDM) is a way to express verifiable electronic attestation of attributes on the Web. At its core, a W3C Verifiable Credentials (VC) is a standardized digital format for presenting and exchanging verifiable claims (in essence statements expressed using subject-property-value relationships) about individuals, organizations, or things. These claims can be expressed as attributes in an electronic attestation of attributes. Specifically designed for the Web, the W3C VCDM aims to enable users to present attribute assertions from potentially different issuers and about potentially different identity subjects. These assertions can be organized into information graphs expressing subject-property-value relationships (e.g. Credential-type-DrivingLicense).

The W3C Verifiable Credentials Data Model (VCDM) is an open standard and is designed to be interoperable across different systems and platforms and to support a wide range of applications. The W3C VCDM v1.1 [i.128] describes a issuer-holder-verifier based model for digital "verifiable credentials" (defined as a "set of one or more claims made by an issuer" that are also "tamper-evident [with] authorship that can be cryptographically verified"). Specifically, the VCDM v1.1 aims to improve the ease of expressing digital credentials while also ensuring a high degree of privacy.

EXAMPLE: A trusted authority, such as a PID Provider, could construct a W3C VCDM compliant attestation containing the PID attributes and sign these with their private key. The user (assumed herein to be the identity subject of the VC) can then create a Verifiable Presentation (VP) using one or more VCs and present attributes to a verifier. The resulting W3C VC is verifiable to any verifier who has access to the required cryptographic keys. The proof mechanism could then support privacy features such as selective disclosure and/or unlinkable verifiable presentations.

After the publication of VCDM v1.1, the W3C VC WG has been working on VCDM 2.0 to make the standard more flexible and able to support multiple formats and signature algorithms. Work was ongoing to support the representation of verifiable claims in multiple ways including JSON, JSON-LD, or using any other data representation syntax capable of expressing the data model such as XML, YAML, or CBOR, as long as there is a mapping defined back to the base data model defined in the VCDM document (which relies on JSON-LD). This work was ongoing as several outstanding issues remained unsolved.

However, recently the W3C VC WG has argued strongly in favor of removing securing JSON and non linked data formats from the specification (see W3C VC WG issue #88 [i.124]). This means that the W3C VCDM is likely to evolve in a direction that will not address outstanding issues with the underspecified JSON sections, which includes key details such as how to do the required transformations or mappings. By extension, it is likely also that the proposed W3C work on how to secure a (W3C) VC using JSON [i.84] will be postponed until further notice. It is worth noting that the W3C VC WG charter does not specify specific media types, but that there does not exist a consensus with the WG to pursue JSON.

Regardless of the debate outcome, each VC and VP includes fields for specifying the signature schemes used to sign the claim or the presentation of a claim respectively (i.e. whether the verification of the proof is calculated against the data transmitted or against a transformation such as another data model or an information graph). Since the debate outcome is presently unknown, the text herein describes the solutions presently mentioned by VCDM v1.1, which are JSON Web Token and Data Integrity Proofs. Each will be described, with illustrations for possible solutions to still outstanding issues for the JWT based approach. The data integrity proofs will only be briefly explained to help readers understand why some of the ideological differences may make it difficult to secure a W3C VC using SD-JWT without a proper specification on how to secure a W3C VC using JSON. Finally, the potential of relying only on SD-JWT will be discussed as it represents the most suitable selective disclosure alternative for the ARF considering the ongoing debates.

7.3.2.2 W3C VC, JSON-LD, data integrity proofs, and linked data signatures

There are many concepts surrounding the W3C VCDM v1.1, including JSON-LD, data integrity proofs, and linked signatures. The first, JSON-LD, will be explained in detail below, but it is helpful to explain how the other two relate to JSON-LD.

Data integrity proofs are defined by the W3C as "a set of attributes that represent a digital proof and the parameters required to verify it." Put differently, a data integrity proof provides information about the proof mechanism, parameters required to verify that proof, and the proof value itself. This information is provided using Linked Data vocabularies in a JSON-LD formatted attestation.

Linked data signatures, is a proposed way to sign data expressed in linked data formats such as a JSON-LD. Linked data signatures sign the underlying information graph as opposed to the payload itself. More specifically, the graph is normalized into a byte stream that is signed. The corresponding verification can be of the graph of information, and not necessarily the syntax specific content itself meaning that the same digital signature would validate information expressed in multiple compatible syntaxes without necessitating syntax specific proofs (see W3C VC Data Integrity v1.0 where this idea is explored in detail).

To understand how a W3C VCDM v1.1 compliant attestation would look like, it is necessary to understand its core format, JSON-LD. Being similar to JSON, a key difference is that JSON-LD uses a property called `"@context"` to link attributes to descriptions that provide semantic clarity on how to unambiguously interpret each attribute. Each attribute is expressed in the form of subject-predicate-object triples that essentially describe an information graph.

Consider the following example of an JSON-LD document describing a person. The attributes `name` and `jobTitle` are mapped to concepts in the schema.org vocabulary as detailed in the `"@context"`.

```
{
  "@context": "http://schema.org/",
  "@id": "https://me.example.com",
  "@type": "Person",
  "name": "John Doe",
  "jobTitle": "ETSI TR editor"
}
```

The @context allows the JSON-LD to be mapped to an Resource Description Framework (RDF) model and thus an information graph. The information graph for the above looks as follows:

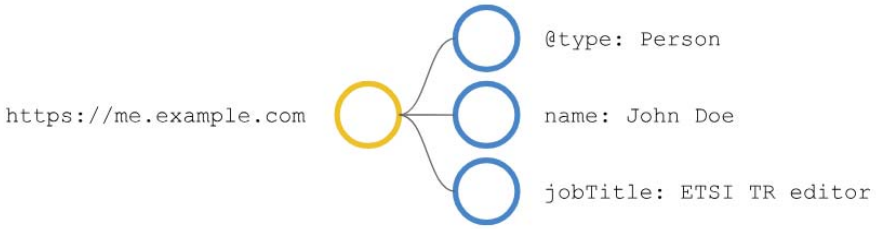

Figure 23: Example of W3C VCDM v1.1 graph

And the W3C VCDM v1.1 graph triples are as follow:

Table 3: Example of W3C VCDM v1.1 graph triples

Subject	Predicate	Object
https://me.example.com	http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://schema.org/Person
https://me.example.com	http://schema.org/jobTitle	ETSI TR editor
https://me.example.com	http://schema.org/name	John/Jane Doe

And the associated N-Quads (a syntax for RDF datasets) are:

1) <https://me.example.com> <http://schema.org/jobTitle> "ETSI TR editor" .

2) <https://me.example.com> <http://schema.org/name> "John/Jane Doe".

3) <https://me.example.com> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://schema.org/Person>.

The benefit with the above is that it does not matter what syntax is used to describe the underlying information graph as they would all describe the same model and thus enable a mapping to the exact same N-Quads.

NOTE: Since data integrity proofs sign the N-Quads containing triples as opposed to only the object, they do not fully support predicates that rely on the algebraic manipulations of the object. For instance, while it is possible to check for message equality, it is not possible to check whether one value is larger than another.

To enable selective disclosure of a W3C VCDM v1.1 using data integrity proofs and linked data proofs, an issuer would need a proof mechanism that can logically order the N-Quads in such a way that the verifier knows that the presented attributes are properly paired. One way is to use the N-Quad message digests as leaf nodes to a Merkle tree and include the Merkle root in the attestation. Another, assuming that the issuer is comfortable with using JSON-LD and linked data proofs only, is to include N-Quad messages as selectively disclosable values in a SD-JWT `"_sd"` array (see clause 7.3.1.2 for a detailed description of how to generate a disclosure in [i.76] (IETF OAUTH: "Selective Disclosure for JWTs (SD-JWT)") and let the user present only the parts of the information graph that the verifier needs. To date, the most well developed solution relies on the `bbs-2022` cryptosuite, which supports JSON-LD + data integrity proofs + linked data proof. And including triples in SD-JWT is not entirely straight forward and would require additional specification.

To conclude, JSON-LD is a way to express linked data and JSON-LD based attestations may include data integrity proofs that also rely on linked data for their verification. When also using linked data proofs, issuers can issue (Q)EAAs that are highly optimized for semantic interoperability. However, it is not entirely clear how selective disclosure and predicates would work in the context of PID/(Q)EAAs. Supporting crypto suites like `bbs-2022` are based on primitives that the public sector is unlikely to use since they are not considered as being plausible quantum safe. Solutions like SD-JWT can support linked data proofs but it is not entirely clear how they could be combined with data integrity proofs (and what the benefits would be) as SD-JWT was designed with JWT based attestations in mind.

Having described how W3C VCDM v1.1 compliant attestations can be secured using SD-JWT also for JSON-LD and linked data signatures, attention now turns to JWT based W3C VCs and SD-JWT.

7.3.2.3 JWT based W3C VC

One popular proof format that is actively used in several implementations is the JSON Web Token (IETF RFC 7519 [i.82]). A JWT encodes claims as a JSON object contained in a JSON Web Signature (JWS) (IETF RFC 7515 [i.80]) or JWE (IETF RFC 7516 [i.81]). A user could present a VP with the VC claims using JWT as described in example 32 of the W3C VC Data Model [i.128]. The decoded JWT contains the presentation as exemplified next.

```
{
   ...,
   "verifiableCredential":[
"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6ImRpZDpleGFtcGxlOmFiZmUxM2...QGbg"
   ]
}
```

The VC contained within (highlighted above in yellow) contains the following information about the identity subject.

```
{
   ...,
   "credentialSubject":{
      "degree":{
         "type":"BachelorDegree",
         "name":"<span lang='fr-CA'>Baccalauréat en musiques numériques</span>"
      }
   }
}
```

The VC contains the attribute in cleartext. Typically, a signed JWT containing identity data cannot support use cases where the JWT is issued once and then presented multiple times by the user who seeks to disclose only the attributes necessary for the service. In and of itself, the W3C VC standard only supports, but does not enforce, selective disclosure by design. The standard is flexible and supports multiple selective disclosure techniques. However, until recently these selective disclosure techniques have relied on multi-message signature schemes like `bbs-2022` suite.

NOTE: The text below assumes that there is a way to secure JSON for W3C VCDM v1.1 and ignores the ongoing debate on the topic within the W3C VC WG.

7.3.2.4 SD-JWT based attestations

To support selective disclosure in JWTs, Fett, Yasuda, and Campbell (2023) specify Selective Disclosure JSON Web Token (SD-JWT) in the Internet Engineering Task Force (IETF) draft document [i.76] entitled "Selective Disclosure for JWTs (SD-JWT)". At its core, an SD-JWT is a digitally signed JSON document that can contain salted attribute digests that the user can selectively disclose using disclosures that are outside the SD-JWT document. This allows the user to share only those PID attributes that are strictly necessary for a particular service.

NOTE 1: SD-JWT is generally applicable to selective disclosure of JWTs that are not bound to the W3C VCDM v1.1. A W3C VCDM v1.1 contains sections that describe how a VC can be JSON encoded in a JWT and then protected using JWS/JWE. Correspondingly, the SD-JWT specifies how any JWT can support selective disclosure.

NOTE 2: An SD-JWT supports selective disclosure solutions that require a clear logical ordering of data. It does not support algebraic manipulations of data.

Each SD-JWT contains a header, payload, and signature. The header contains metadata about the token including the type and the signing algorithm used. The signature is generated using the PID Provider's private key. The payload includes the proof object that enables the selective disclosure of attributes. Each disclosure contains a salt, a cleartext claim name, and a cleartext claim value. The issuer then computes the hash digest of each disclosure and includes each digest in the attestation it signs and issues.

Using the proof object and the user shared disclosures, the verifier can verify that the disclosed claims were part of the original attestation. To do so, the verifier first verifies the issuer's signature over the entire SD-JWT. The verifier then calculates the digest over the shared disclosures and checks that the digest is included in the signed SD-JWT. Since the SD-JWT includes only digests of disclosable attributes, the verifier can only learn about claim names and claim values that are disclosed by the user or that are included as clear-text claims. The verifier cannot learn about any other claim names or values as these are included in the SD-JWT as salted attribute digests.

The IETF SD-JWT draft specification v0.4 [i.76] of 2023-04-11 details the exact process of creating a disclosure in section 5.1.1.1. In essence, for each disclosable claim, the issuer generates and associates a random salt with each key value pair, and encodes the byte representation of these as base64url. An example of a disclosure is shown in Figure 24.

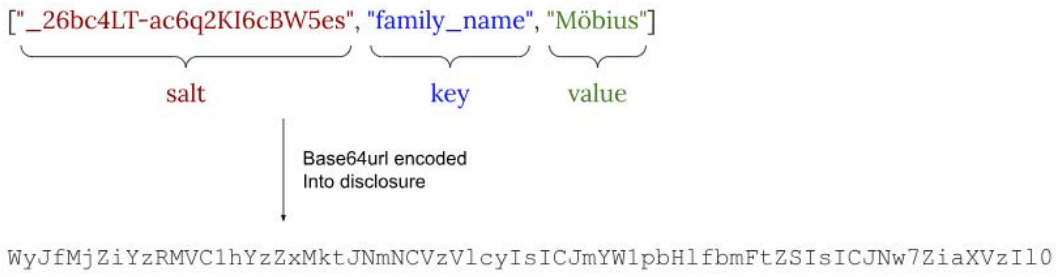

Figure 24: Example of SD-JWT disclosure

Figure 24 illustrates an example with the byte representation of the JSON-encoded array containing the salt, key, and value, is base64url-encoded into the disclosure.

NOTE 3: A linked data signature could be included in the `_sd` array but it is not entirely clear how to handle triples in the disclosure. One option could be: `[<salt>, <subject-predicate>, <object>]`

To embed a disclosure in the SD-JWT, the issuer hashes each disclosure using a specified hash algorithm. The base64url encoded bytes of the digest, and not the disclosure, is then included in the SD-JWT as an array in the claim `_sd`, which includes only an array of strings, each being the digest of a disclosure or a random number (used to hide the original number of disclosures). This array is randomized so that the order of attribute disclosures is not always the same.

The SD-JWT specification supports selectively disclosable claims in both flat and more complex nested data structures. The issuer can therefore decide for each key individually, on each level of the JSON, whether or not the key should be selectively disclosable. The `_sd` claim is included in the SD-JWT at the same level as the original claim. Selectively disclosable claims can in turn include other objects with selectively disclosable claims.

Below, this text only exemplifies the flat and the nested data structure examples, but others are possible too.

Table 4: Example of SD-JWT using a flat data structure

Contents	["imQfGj1_M0El76kdvf7Daw", "address", {"street_address": "Schulstr. 12", "locality": "Schulpforta", "region": "Sachsen-Anhalt", "country": "DE"}]
Disclosure	WyJpbVFmR2oxX00wRWw3NmtkdmY3RGF3IiwgImFkZHJlc3MiLCB7InN0cmVldF9hZGRyZXNzIjogIlNjaHVsc3RyLiAxMiIsICJsb2NhbGl0eSI6ICJTY2h1bHBmb3J0YSIsICJyZWdpb24iOiAiU2FjaHNlbi1BbmhhbHQiLCAiY291bnRyeSI6ICJERSJ9XQ
Digest	FphFFpj1vtr0rpYK-14fickGKMg3zf1fIpJXxTK8PAE
_sd value	{ "_sd": ["FphFFpj1vtr0rpYK-14fickGKMg3zf1fIpJXxTK8PAE"], ..., "_sd_alg": "sha-256" }

Table 5: Example of nested SD-JWT with the sub-claim country in cleartext

Contents	["QSNIhu_n6a1rI8_2eNARCQ", "street_address", "Schulstr. 12"], ["QPkblxTnbSLL94I2fZIbHA", "locality", "Schulpforta"], ["jR-Yed08AEo4gcogpT5_UA", "region", "Sachsen-Anhalt"]
Disclosures	WyJRU05JaHVfbjZhMXJJOF8yZU5BUkNRIiwgInN0cmVldF9hZGRyZXNzIiwgIlNjaHVsc3RyLiAxMiJd, WyJRUGtibHhUbmJTTEw5NEkyZlpJYkhBIiwgImxvY2FsaXR5IiwgIlNjaHVscGZvcnRhIl0, WyJqUi1ZZWQwOEFFbzRnY29ncFQ1X1VBIiwgInJlZ2lvbiIsICJTYWNoc2VuLUFuaGFsdCJd
Digests	"G_FeM1D-U3tDJcHB7pwTNEElLal9FE9PUs0klHgeM1c", "KlG6HEM6XWbymEJDfyDY4klJkQQ9iTuNG0LQXnE9mQ0", "ffPGyxFBnNA1r60g2f796Hqq3dBGtaOogpnIBgRGdyY"
_sd value	{ "address": { "_sd": ["G_FeM1D-U3tDJcHB7pwTNEElLal9FE9PUs0klHgeM1c", "KlG6HEM6XWbymEJDfyDY4klJkQQ9iTuNG0LQXnE9mQ0", "ffPGyxFBnNA1r60g2f796Hqq3dBGtaOogpnIBgRGdyY"], "country": "DE" }, ..., "_sd_alg": "sha-256" }

The QTSP/PIDP will have to send the raw claim values contained in the SD-JWT, together with the salts, to the EUDI Wallet user. The SD-JWT standard requires that data format for sending the SD-JWT and the disclosures to the EUDI Wallet user is a series of base64url-encoded values in what is called the Combined Format for Issuance, which looks like follows: `<SD-JWT>~<Disclosure 1>~<Disclosure 2>~...~<Disclosure n>~<optional Holder Binding JWT>`. Note the separation of between the values using ~.

When the EUDI Wallet user receives the attestation from the QTSP/PIDP, the SD-JWT standard requires that the user verifies the disclosures. The user does so by extracting the disclosures and the SD-JWT from the Combined Format for Issuance, hashing each disclosure, and accepts the SD-JWT only if each resulting digest exists in the `_sd` array.

Relatedly, during presentation, the user sends the SD-JWT and the *n* disclosures to the verifier as a series of base64url encoded values in what is called the Combined Format for Presentation, which looks as follows: `<SD-JWT>~<Disclosure 1>~<Disclosure 2>~...~<Disclosure n>~<optional Holder Binding JWT>`

The verifier checks that the issuer's signature is valid over the SD-JWT, that the disclosure digests are part of the SD-JWT, and if applicable that the Holder binding is valid (for specific steps see clause 6.2 in the present document).

Having described JSON secured W3C VCs and how SD-JWT can ensure selective disclosure of JWT based attestations, the text next discusses the potential joint utilization of both W3C VCs and SD-JWT, and why it is not as straightforward as it may appear.

7.3.2.5 Securing the W3C VC payload using SD-JWT

It is very difficult to clearly communicate options on how to secure a W3C VC using SD-JWT given the two main ways a W3C VC can be secured and given the lack of agreement on whether or not to secure JSON within the W3C VC WG, and given how SD-JWT was designed with JWT based attestations in mind. As such, the text herein is speculative.

The focus of SD-JWT is to specify how claims in a JWT can be selectively disclosed. This applies to any type of attestation where attribute assertions are JSON encoded in a JWT, including potential JWT versions of any W3C VCDM v1.1 compliant attestation (assuming that future work in the W3C VC WG will also secure JSON).

The April 11 specification of SD-JWT includes an appendix that exemplifies how to use the SD-JWT specification to secure a payload represented as a W3C VC data model. Relatedly, the W3C VCDM recommendation contains examples of W3C VCs encoded as JWT. However, the two examples build on different assumptions. One way to jointly utilize W3C VCDM v1.1 and SD-JWT is to include the entire W3C VC as a claim value in the SD-JWT. Another way is to rely on a transformation algorithm that would allow a verifier to recreate the W3C VC from an SD-JWT that uses JSON only. Both have their associated challenges.

The SD-JWT specification does support selective disclosure of a W3C VCDM v1.1 compliant attestation either as an embedded value, e.g. as `"vc": {<W3C VC>}`, or using a transformation algorithm (for an example using VCDM 2.0 see clause 9.1 of [i.128]). Similarly, it is possible to rely on proposals similar to the W3C Securing Verifiable Credentials using JSON Web Tokens [i.125] and use SD-JWT to secure it.

Relatedly, the VCDM v1.1 introduces one way to design VCs that could be jointly utilized with SD-JWT. The VCDM v1.1. uses a JWT to secure a VC payload that needs to follow the rules for a JSON-LD payload. Consequently, the JWT is an envelope, which means that it is not entirely compatible with more recent drafts of SD-JWT. There is also confusion on how to include JWT claims in the credential payload. Furthermore, the presentation is another JWT, where the VC is embedded. Such a design is not without problems.

NOTE 1: Until recently, the VCDM 2.0 included proposals that would address limitations in the VCDM v1.1. These proposals in the VCDM 2.0. would require only that the VC can be mapped into a JSON-LD representation (can be one directional). Consequently, a VC can be just a JWT or a SD-JWT using a pure JSON payload. The way presentations are created is also up to the respective presentation. However, the ongoing disagreement around the continued support for this work (see W3C VC WG issue #88 [i.124]) means that it is no longer clear that the W3C VCDM 2.0 will support JSON. And since W3C VCDM v1.1 requires additional work to fully work with SD-JWT, the way to secure an W3C VCDM v1.1 compliant attestation using SD-JWT is unclear.

NOTE 2: The SD-JWT specification published on April 11 2023 is developed around the assumption that the VCDM 2.0. would secure JSON too. Relatedly, the ARF 1.0 text mandates VCDM v1.1. compliance with the assumption that there would be a way to rely on pure JSON payloads (the Type 1 configuration in ARF 1.0 mandates JSON and not JSON-LD).

To exemplify possible joint utilizations, the following VCDM v1.1. compliant attestation will first be populated with some of the mandatory PID attributes. The example will utilize an external proof since data integrity proofs are of questionable use in the PID context (which means that it does not require a `proof` property). The content is shortened for brevity and only includes values relevant for selective disclosure.

```
{
  "@context":[
  <>
  ],
  "id":"http://example.com/credentials/4643",
  "type":[
    "VerifiableCredential",
    "IdentityCredential"
  ],
  "issuer":"https://example.com/issuers/14",
  "issuanceDate":"2010-01-01T19:23:24Z",
  "credentialSubject":{
    "givenName":"Jane",
    "familyName":"Doe",
    "birthDate":"2000-01-01"
  }
}
```

Figure 25: Example of W3C VC with some ARF 1.0 mandatory PID attributes

The example in Figure 25 shows a W3C VC Data Model 1.1 compliant attestation with some of the mandatory PID attributes as mentioned in ARF 1.0. The identity data are highlighted in blue.

To secure the above attestation with a JWT and enable selective disclosure, it is necessary to create a disclosure of each mandatory attribute claim in the `credentialSubject` property (using the method shown above in clause 7.3.2.3), and then to create a valid JWT. This may appear to be straightforward, but the issuer needs to decide:

1) whether or not to use linked data proofs; and

2) whether or not to use the SD-JWT as a container; or

3) rely on a transformation algorithm.

To discuss every possible option is outside the scope of this text; only a single option is shown for illustrative purposes.

One possible way is to put the credential payload `"vc"` claim to differentiate it from the claims in the JWT that is used as the security envelope. Furthermore, because JWT uses different property names, some implementations duplicate the `iss`, `jti`, and `iat` claim names while others rely on the mapping proposed in the JSON encoding section in the W3C VCDM v1.1 recommendation. Below, the example uses the duplicate claim names because this is how the examples are provided in the W3C VCDM v1.1 recommendation (duplicate claim names are optional in IETF RFC 7519 [i.82]). Note the omission of the `sub` claim due to it being selectively disclosable. Finally, the proof is omitted in Figure 26.

```
{
  "vc":{
    "@context":[
      "..."
    ],
    "id":"http://example.com/credentials/4643",
    "type":[
      "VerifiableCredential",
      "IdentityCredential"
    ],
    "issuer":"https://example.com/issuers/14",
    "issuanceDate":"2010-01-01T00:00:00Z",
    "credentialSubject":{
      "_sd":[
        "2cj...szs",
        "H03...iVY",
        "S7e...uDc"
      ]
    },
    "_sd_alg":"sha-256"
  },
  "iss":"https://example.com/issuers/14",
  "jti":"http://example.com/credentials/4643",
  "iat":"1262304000"
}
```

Figure 26: Example of how SD-JWT could secure a W3C VC

The example in Figure 26 shows a possible way an SD-JWT could secure an W3C VC Data Model 1.1 compliant attestation containing the mandatory PID attributes as disclosure digests (highlighted in blue). Conflicts that exist between the W3C VC Data Model 1.1 and SD-JWT were resolved by adhering to the W3C standard. Appendix A.4. from the SD-JWT specification draft 4 was used as the basis for this example.

A Verifiable Presentation for the above VC looks as follows:

```
{
  "iss": "some key identifier",
  "aud": "did:example:4a57546973436f6f6c4a4a57573",
  "nbf": 1541493724,
  "iat": 1541493724,
  "exp": 1573029723,
  "nonce": "343s$FSFDa-",
  "vp":{
    "@context": [
      "https://www.w3.org/2018/credentials/v1",
      "https://www.w3.org/2018/credentials/examples/v1"
    ],
    "type": [
      "VerifiablePresentation"
    ],
    "verifiableCredential": [
      "..."
    ]
  }
}
```

Figure 27: Example of a VP for an SD-JWT secured W3C VC

The example in Figure 27 shows a Verifiable Presentation for an SD-JWT secured W3C Verifiable Credentials Data Model v1.1 [i.128] compliant attestation. The SD-JWT is a base64url encoded string.

There are some difficulties with using an SD-JWT (the IETF SD-JWT draft specification v0.4 [i.76]) and the W3C Verifiable Credentials Data Model v1.1 [i.128] as illustrated in the examples above. Some important difficulties are:

- A lack of a syntax definition catering for the selective disclosure capability in SD-JWT. Put differently, it is possible to include the digests of the disclosures and decoys in the VC but it is not entirely clear how this would harmonise with linked data principles. It is also unclear how the presentation of selectively disclosable attributes will look like. The W3C VCDM was designed with presentation capabilities in mind where attributes from multiple VCs, about potentially different subjects, could be combined into a single presentation. In contrast, the SD-JWT combined presentation format is focused on ease of use and ease of deployment.

- Selectively disclosable claims are base64url encoded twice (once in the SD-JWT and once again in the VC. This double encoding adds inefficiencies.

- There exists confusion in how to use preexisting JWT claims, e.g. `sub`, in the credential payload. Some follow the duplicate claims approach (which is prevalent in the examples in the W3C VCDM v1.1 text). Others rely on the provided JSON encoding rules and the W3C VCDM v1.1 implementer's guidelines recommendations.

- JSON-LD was not designed to extend into the SD-JWT. The interaction between the JSON-LD context and the disclosures protected in the SD-JWT is undefined until after the claims have been decoded from the SD-JWT (assuming the context defines the disclosable attributes and not the selective disclosure array).

One may try different versions of their joint utilization to circumvent some of the four above mentioned problems. But perhaps most importantly, the above example may trigger questions as to the benefits derived from combining JSON-LD with SD-JWT. The former was developed to ensure semantic interoperability in an open data world. And SD-JWT was designed to provide selective disclosure capabilities to a JSON based attestation in a JWT. Using JSON-LD for W3C VC together with data integrity proofs provide benefits in a usage scenario where the actors face semantic interoperability challenges that can be resolved by accessing other related information about a particular thing. Seemingly, jointly utilizing JSON-LD based W3C VCs with SD-JWT does not result in their combined benefits, but rather that their respective benefits are not utilized.

One alternative is to simply use SD-JWT only, also for W3C VCDM v1.1 attestations, and rely on transformation algorithms to re-create the W3C VC.

7.3.2.6 Using SD-JWT only

The SD-JWT specification provides a format that is optimized for the transport of the credential including the disclosures without further encoding. It is not designed to be embedded into any envelopes. It is arguably better to simply rely on JSON only claims for SD-JWT and recreate the W3C VC. An example is provided next.

```
{
  "alg": "ES256",
  "typ": <>,
  <other header info>
}
.
{
  "iss":"https://example.com/issuers/14",
  "nbf":"1262304000",
  "iat":"1262304000",
  "type": "eu.europa.ec.eudiw.pid.se.1",
  "_sd":[
    "2cj...szs",
    "H03...iVY",
    "RKE...omY",
    "S7e...uDc"
   ],
   "_sd_alg":"sha-256"
}
```

Figure 28: Example of a SD-JWT compliant with the W3C VCDM

The example in Figure 28 shows an SD-JWT secured attestation (not using JSON-LD) with the mandatory and disclosable PID attributes highlighted in blue. The `"_sd"` is here included as a root claim. This SD-JWT can be consumed, without prior processing, by any compliant SD-JWT library. Further evaluation can be done using standard JWT payload processing algorithms. In the example in Figure 28.

- The JOSE header indicates the type, which could include W3C VC with pure JSON payload.
- The claims in the credential are standard JWT claims. Applications can use predefined and established JWT claims from the "JWT Claims Registry", like `"sub"` for user identifiers. They can also use more complex claim structures such as those defined by OpenID Connect for Identity Assurance for providing information about provenance and level of assurance. This means existing JWT-based implementations can consume such VC payloads directly.
- The type communicates to the verifier how to interpret any disclosed claim and there is no need for a separate `@context`.

A presentation is constructed using the combined format for presentation as defined in the SD-JWT specification.

NOTE: The present document recommends using the IETF April 11 version of SD-JWT without Appendix A4 and A5 to understand the selective disclosure mechanism. Relatedly, to understand how to use SD-JWT as an attestation format, see the 2023-06-17 version of "SD-JWT-based Verifiable Credentials with JSON payloads (SD-JWT VC)" [i.75].

7.3.2.7 SD-JWT and multi-show unlinkable disclosures

Because every SD-JWT disclosure contains a unique salt, this unique salt acts as an identifier for the entire SD-JWT. Put differently, it is enough for a malicious issuer to receive a single disclosure from a colluding verifier for the issuer to uniquely identify the identity subject. Similarly, colluding verifiers could compare salt values to link together presentations from the same user (see clause 9.4 in the SD-JWT [i.76] specification for additional details).

While it is entirely impossible to prevent issuers from identifying the user based on the unique salt in the salted hash digests approach, it is possible to enable multi-show unlinkable disclosures even if verifiers collude or if a single curious verifier attempts to learn more about the user than what is disclosed in each presentation. To achieve complete multi-show unlinkability it is required that:

1) each SD-JWT contains only unique salts (even for the same claim); and

2) each SD-JWT is associated with a unique cryptographic key material used for device binding and/or holder binding (denoted as "holder binding key" in the context of SD-JWT).

Consequently, issuers are required to rely on batch issuance of SD-JWT to the EUDI Wallet if device retrieval functionality is desired (in an online scenario, the user can request a new SD-JWT on demand).

NOTE: To reduce the burden on issuers, it is possible to introduce a limit on the number of uses of each SD-JWT. The user would then be linkable in a portion of their presentations.

EXAMPLE: A user is given 10 PID attestations as SD-JWT. The user presents the first 9 SD-JWTs once and the 10th twice. Out of the 11 presentations, two are linkable.

7.3.2.8 Predicates in SD-JWT

Similar to MSO, an SD-JWT was not designed to support predicates that can be dynamically computed (e.g. to compute an age over proof from the birth date). Here too, the recommendation is to use static claims with boolean values such as `"age_over_NN": "True"`. However, as presented above in clause 4.5.2, it is possible to rely on issuer signed computational inputs and parameters to enable dynamic predicate support in SD-JWT.

7.3.3 Analysis of using SD-JWT as attestation format applied to eIDAS2

An analysis of the W3C VC and IETF SD-JWT formats applied to an eIDAS2 context results in the following observations and recommendations:

- The W3C VC Data Model v1.1 in conjunction with IETF SD-JWT should be supported by an EUDI Wallet according to the ARF 1.0 [i.34]. However, this is problematic given the difficulties detailed in clause 7.3.2. Consequently, the recommendation of the present document is to use SD-JWT only and to rely on transformation algorithms if issuers want to support W3 VCDM v1.1.
- It is possible to jointly utilize JSON-LD and linked data proofs with SD-JWT, but data integrity proofs remain an open question.
- The present document recommends using SD-JWT as a standalone attestation format.
- The SD-JWT issuer corresponds to a QTSP and/or a PIDP.
- The SD-JWT verifier corresponds to an eIDAS2 relying party (that will validate the SD-JWT as a (Q)EAA/PID).
- The eIDAS2 relying party should use the eIDAS2 EU TL to retrieve the QTSP/PIDP trust anchor.
- The eIDAS2 relying party should validate the attestation (submitted by the EUDI Wallet) according to the principles described in clause 7.3.2; the issuer's signature should be validated by using the QTSP/PIDP trust anchor.
- The SD-JWTs in the EUDI Wallet should all use unique salts as described in clause 7.3.2 to cater for unlinkability when validated by the relying party.
- The SD-JWT is signed by the QTSP/PIDP with a JOSE formatted signature, which allows for SOG-IS approved cryptographic algorithms [i.115] and for QSC for future use [i.73].
- The SD-JWT may be signed with an ETSI JAdES signature if supported by the relying party. Thus, the JAdES signature format may contain additional information about revocation information, CA-chains and time-stamps.

These observations and recommendations should be considered with respect to selective disclosure for the ETSI work items ETSI TS 119 462 [i.48], ETSI TS 119 471 [i.49] and ETSI TS 119 472 [i.50].

7.4 Secure storage of mDL/VC and keys in EUDI Wallet

The ISO mDL authentication key and SD-JWT holder binding keys should be protected in the device's Trusted Execution Environment (TEE) or a Secure Element (SE); the user should be able to access the ISO mDL authentication key and SD-JWT holder binding key by authentication with a PIN-code or the use of biometrics. Several ISO mDL and SD-JWT data elements are PII and should therefore be stored securely. Encryption at rest of the SD-JWT is recommended, and if possible the SE/TEE should be used to perform the encryption, with keys protected by the SE/TEE, or else the ISO mDL and SD-JWT should be stored in the SE/TEE.

The ARF [i.34], clause 6.5.3 and table 5 also specify how to store and access the PID/(Q)EAA cryptographic keys in a device used by the EUDI Wallet.

From a regulatory perspective, the eIDAS2 [i.54] article 6c specifies the legal requirements on an EUDI Wallet certification, which will be defined in a CIR (Commission Implementing Regulation). This CIR will in turn refer to ENISA's EUCC (EU Cybersecurity Certification scheme), which may regulate the certification requirements on protection of the PID/(Q)EAA as ISO mDL and SD-JWT.

Furthermore, CEN TC/224 WG17 may specify Common Criteria Protection Profiles (CC PP) on how to protect the PID/(Q)EAA and associate cryptographic keys related to the ENISA EU-CC; such EUDI Wallet CC PP may be based on TC/224 WG17 [i.27]. Also, TC/224 WG20 [i.28] are specifying how to onboard the PID to an EUDI Wallet, which involves the associated cryptographic key protection as well.

Other certification standards that may underpin the ENISA EU-CC scheme are Global Platform TEE Protection Profile [i.61] and Eurosmart PP-0117 Protection Profile for Secure Sub-System in System-on-Chip (3S in SoC) [i.56].

Additional recommendations on how to store and protect credentials and the associated cryptographic keys in a digital wallet are available in the the DIF Wallet Security [i.39], ISO/IEC 23220-6 [i.90] and W3C Universal Wallet [i.126] specifications.

NOTE: Complete descriptions about storage of PID/(Q)EAA, protection of cryptographic keys and EUDI Wallet certifications go beyond the scope of the present document, but an overview is provided in the present clause since the cryptographic keys are of relevance to selective disclosure of PID/(Q)EAA in the formats of ISO mDL and SD-JWT.

8 Conclusions

The eIDAS2 regulation and the Architecture Reference Framework (ARF) define regulatory requirements on selective disclosure and unlinkability for the EUDI Wallet. The present ETSI technical report provides a comprehensive analysis of signature schemes, credential formats and protocols that cater for selective disclosure, unlinkability, predicates and Zero-Knowledge Proofs.

Several of the analysed signature schemes are designed for specific use cases, such as cryptocurrencies. For example, some of the zk-SNARK protocols have been designed to allow for Zero-Knowledge Proofs of the cryptocurrency protocol ZeroCash. In addition to this, Zero-Knowledge Proofs are strictly speaking not required to meet the regulatory requirement set forth in the eIDAS2 regulation and the ARF.

Furthermore, several signature schemes that cater for selective disclosure are based on pairing-based elliptic curve cryptographic algorithms that are not approved by SOG-IS and are not considered as plausible quantum-safe. For example, the BBS signature scheme is such a scheme. Hence, these signature schemes cannot be used by the EUDI Wallet for the public sector in the EU.

More specifically, the present document analyses the ISO mobile driving license (ISO mDL), W3C Verifiable Credentials (VCs) in conjunction with SD-JWT, and SD-JWT as standalone, since these credentials formats are relevant as Person Identification Data (PID) and Qualified Electronic Attestation of Attributes (QEAAs) for the EUDI Wallet.

The ISO mDL and SD-JWT formats and related presentment protocols cater for selective disclosure and unlinkability based on the concept of hashes of salted attributes. Furthermore, these credential formats support SOG-IS approved cryptographic algorithms and can also be used with quantum-safe cryptography for future use. The conclusion is thus that ISO mDL and SD-JWT meet the eIDAS2 regulatory and technical requirements on selective disclosure, unlinkability and cryptographic algorithms. The present document also proposes a new approach to calculate predicates based on hash chains in conjunction with hashes of salted attributes, which can be used for dynamically deriving statements about the user without revealing the attribute values.

The present document gives recommendations on how eIDAS2 compliant QTSPs should issue PID/(Q)EAAs in the form of ISO mDL and/or SD-JWT that cater for selective disclosure. For use cases that require W3C VCDM v1.1 compliant attestations, the present document recommends using a transformation algorithm to recreate the original W3C VC. The present document notes that SD-JWT can provide selective disclosure capability also for attestations that use JSON-LD and linked data proofs, but that support for data integrity proofs is questionable.

In order to achieve unlinkability, the random salts in the ISO mDL MSO and SD-JWT should be unique, meaning that refreshed MSOs and SD-JWTs are presented to a relying party. There are many similarities between the ISO mDL issuers and the eIDAS2 QTSPs or PID providers, which could be harmonised in ETSI TS 119 471 [i.49] and ETSI TS 119 472 [i.50] that will standardize the issuance policies and profiles of (Q)EAAs. More specifically, the ISO mDL MSO could be issued by an eIDAS2 QTSP certification authority, meaning that the EU trusted lists can be used to retrieve revocation information and trust anchors when validating the ISO mDL MSO signature. ETSI TS 119 495 [i.46], which specifies certificate profiles and TSP policies for Open Banking and PSD2, may partially be re-used for the issuance of ISO mDLs as (Q)EAAs. The same principles could be applied on QTSPs and PID providers that will issue PIDs/(Q)EAAs in conjunction with SD-JWT, although the existing specifications do not specify the issuance policies in detail.

Furthermore, there are recommendations on how to store such credential formats in the EUDI Wallet, and how to present selectively disclosed attributes to eIDAS2 relying parties. The presentation protocols for the ISO mDL and OID4VP/SIOP2 are specified in the ARF, and the present report describes how to use these protocols for selective disclosure of attributes in ISO mDL and SD-JWT.

Annex A:
Comparison of selective disclosure mechanisms

A.1 Selective disclosure signature schemes

Table A.1 provides a comparison of the investigated selective disclosure signature schemes.

Table A.1: Comparison of selective disclosure signature schemes

Signature scheme	Cryptography	Quantum-safe	Unlinkability	Predicates	Reference
Category: Atomic attribute credentials					
Atomic attribute credentials	Conditional: depends on the signature on the credential	Yes	Yes	Conditional: can enroll for atomic attributes with boolean attributes	See clause 4.2
Category: Multi-message signature schemes					
Boneh-Boyen- Shacham (BBS) signatures	Multi-message signature scheme based on ECC bilinear pairings	No	Yes	Yes (in theory)	See clause 4.3.1
Camenisch- Lysyanskaya (CL) signatures	Multi-message signature scheme based on strong RSA assumption	No	Yes	Yes (in theory)	See clause 4.3.2
Mercurial Signatures	Multi-message signature scheme based on decisional Diffie-Hellman (DDH)	No	Yes	Yes (in theory)	See clause 4.3.3
Pointcheval- Sanders Multi-Signatures (PS-MS)	Multi-message signature scheme based on improved CL-signatures	No	Yes	Yes (in theory)	See clause 4.3.4
Category: Hashes of salted attributes					
Salted hashes of attributes	Salted hashes of attributes, signed with RSA, ECC, or QSC	Yes	Conditional: can issue multiple signed objects with salted hashes of attributes that are unique	Conditional: can filter out attributes in a UI, or set boolean attributes in the PID/(Q)EAA, or use HashWires (clause 4.5.2)	See clause 4.4

Signature scheme	Cryptography	Quantum-safe	Unlinkability	Predicates	Reference
Category: Proofs for arithmetic circuits					
Bulletproofs	Proofs for arithmetic circuits based on Fiat-Shamir heuristics	No	Yes	Yes	See clause 4.5.1
zk-SNARK	Proofs for arithmetic circuits based on various mechanisms in Annex A.4	Some zk-SNARK schemes are QSC	Yes	Yes	See clause 4.5.3 and clause A.4
zk-STARK	Proofs for arithmetic circuits based on various mechanisms	Yes	Yes	Yes	See clause 4.5.4

A.2 Credential formats with selective disclosure

Table A.2 provides a comparison of the investigated credential formats with selective disclosure.

Table A.2: Comparison of credential formats with selective disclosure

Credential format	Scheme	Encoding	Maturity	Reference
Category: Atomic attribute credentials				
IETF X.509 attribute certificates	Atomic attribute credentials	ASN.1/DER	X.509 attribute certificate (IETF RFC 5755 [i.78]) is an IETF PKIX standard	See clause 5.2.2
W3C Verifiable Credentials	Atomic attribute credentials	JSON-LD or JWT	W3C VC Data Model [i.128] is a standard	See clause 5.2.3
Category: Multi-message signature schemes				
Hyperledger AnonCreds	CLRSA-signatures	JSON (JWS)	Deployed in Government of British Columbia, IDunion, and the IATA Travel Pass	See clause 5.3.3
W3C VC with CL-signatures	CL-signatures	JSON (LD)	W3C VC Data Model [i.128], implemented in several wallets	See clause 5.3.1
W3C VC Data Integrity with BBS signatures	BBS signatures	JSON (LD)	W3C VC Data Integrity [i.127]	See clause 5.3.2
Category: Hashes of salted attributes				
IETF SD-JWT	Salted hashes of attributes	JSON (JWT)	IETF SD-JWT draft standard [i.76], several reference implementations	See clause 5.4.1
ISO/IEC 18013-5 MSO (Mobile Security Object)	Salted hashes of attributes	CBOR/CDDL (COSE)	ISO/IEC 18013-5 [i.87], implemented in several wallets, deployed in the US	See clause 5.4.2
Category: JSON container formats				
IETF JSON Web Proof	Flexible: CL-signatures, BBS, etc.	JSON (JWS)	IETF JSON Web Proof draft standard [i.67]	See clause 5.5.1
W3C JSON Web Proofs For Binary Merkle Trees	Merkle trees	JSON Web Proofs	W3C draft specification	See clause 5.5.1

A.3 Selective disclosure systems and protocols

Table A.3 provides a comparison of the investigated selective disclosure protocols.

Table A.3: Comparison of selective disclosure systems and protocols

Protocol	Credentials	Protocol	Maturity	Reference
Category: Atomic attribute credentials				
IETF X.509 attribute certificate (protocol)	IETF X.509 attribute certificates	Attribute certificate authorization protocol	X.509 attribute certificate [i.78] is an IETF PKIX standard	See clause 6.2.1
VC-FIDO	W3C Verifiable Credentials	VC-FIDO	Deployed as a prototype at NHS in the UK	See clause 6.2.2
Category: Multi-message signature schemes				
Hyperledger AnonCreds (protocol)	AnonCreds [i.64] based on CLRSA-signatures	Hyperledger Aries protocol [i.65] in conjunction with Hyperledger Indy [i.67] and Hyperledger Ursa [i.68]	Deployed in Government of British Columbia, IDunion, and the IATA Travel Pass	See clause 6.3
Category: Hashes of salted attributes				
ISO/IEC 18013-5 (device retrieval)	ISO/IEC 18013-5 mDL/MSO [i.87]	ISO mDL/MSO over BLE/NFC	ISO standard, implemented in several wallets, deployed in the US	See clause 6.5.2
ISO/IEC 18013-7 (unattended)	ISO/IEC 18013-5 mDL/MSO [i.87]	SIOP2 [i.107], OIDC4VP [i.106]	Draft ISO/IEC 18013-7 [i.88] standard, correlated with ISO/IEC 23220-4 [i.89]	See clause 6.5.4
ISO/IEC 23220-4	ISO mDL [i.87], SD-JWT [i.76], etc.	SIOP2 [i.107], OIDC4VP [i.106]	Draft standard, correlated with ISO/IEC 18013-7 [i.88]	See clause 6.6
Category: Server retrieval flows				
ISO/IEC 18013-5 (server retrieval)	OpenID Connect ID-Token [i.105]	OpenID Connect (OIDC) Core [i.105]	ISO standard, implemented in several wallets, deployed in the US	See clause 6.5.3
Category: ABC (Attribute Based Credentials)				
Idemix	Idemix ABC credentials [i.69] based on CL-signatures	Idemix ABC protocol [i.69]	Implemented by IBM, Hyperledger Fabric [i.66], IRMA project [i.112], and the EU-projects PrimeLife [i.111] and ABC4Trust [i.70]	See clause 6.4
U-Prove	U-Prove ABC credentials [i.100]	U-Prove ABC protocol [i.100]	Implemented in Microsoft's Identity Metasystem and the EU-project ABC4Trust [i.70]	See clause 6.7

A.4 zk-SNARK protocols

Table A.4 provides a comparison of the different zk-SNARK protocols.

The comparison is made based on transparency, universality, and plausible quantum-safety. A transparent protocol is defined as it does not require any trusted setup and uses public randomness. A universal protocol is defined as it does not require a separate trusted setup for each circuit. A plausibly quantum-safe protocol is one that is not considered to be vulnerable to attacks by quantum computing algorithms.

Table A.4: Comparison of zk-SNARK protocols

Protocol	Published	Transparent	Universal	Quantum-safe
Pinocchio [i.108]	2013	No	No	No
Geppetto [i.35]	2015	No	No	No
TinyRAM [i.8]	2013	No	No	No
Buffet [i.119]	2015	No	No	No
ZoKrates [i.40]	2018	No	No	No
xJsnark [i.95]	2018	No	No	No
vnTinyRAM [i.10]	2014	No	Yes	No
MIRAGE [i.94]	2020	No	Yes	No
Sonic [i.97]	2019	No	Yes	No
Marlin [i.32]	2020	No	Yes	No
PLONK [i.59]	2019	No	Yes	No
Spartan [i.99]	2019	No	Yes	Yes
SuperSonic [i.21]	2020	Yes	Yes	No
Hyrax [i.120]	2018	Yes	Yes	No
Halo [i.14]	2019	Yes	Yes	No
Virgo [i.132]	2020	Yes	Yes	Yes
Ligero [i.3]	2017	Yes	Yes	Yes
Aurora [i.9]	2019	Yes	Yes	Yes

Annex B:
Code examples

B.1 Hash chain code example

This annex contains a Python code example of how to use hash chains to calculate a predicate of a user's age.

```
import secrets
from hashlib import sha256

# Get the user's age
while True:
  try:
    age = int(float(input("Enter your age: ")))
    if age < 0:
      raise ValueError
    break
  except ValueError:
    print("Enter a non negative number.")

# The issuer generates  a seed and the commitment the user will need.
seed = secrets.token_bytes()
commitment = sha256(seed)
hash_chain = [commitment.hexdigest().encode('ascii')]

# The issuer then generates the hash chain.
for i in range(age):
  commitment = sha256(commitment.hexdigest().encode('ascii'))
  hash_chain.append(commitment.hexdigest().encode('ascii'))

# The hash chain is reversed so that the index values equal age
hash_chain.reverse()

# The issuer includes the following claim in the signed attestation
age_is_zero = hash_chain[0]

# The verifier wants a proof for age_over_n
n = 10
age_proof = None

# The user has to generate the following age proof
assert isinstance(n, int) and n >= 0, "The value is a non-negative integer."
try:
  age_proof = hash_chain[n] if n != 0 else age_is_zero
  print(f"The proof value is: {age_proof}")
  print(f"Copy this value for the next cell's input prompt: {age_proof.decode('ascii')}")
except IndexError:
  print(f"The user does not have a long enough hash chain for the required age proof of {n}")

# The user sends the age proof to the verifier, who verifies the chain length
age_proof_test = input("Copy paste the provided value from the previous cell: ")
age_proof_test = age_proof_test.encode('ascii')

above_n = False
if n == 0 and age_proof_test == age_is_zero:
  above_n = True
else:
  for i in range(n):
    age_proof_test = sha256(age_proof_test).hexdigest().encode('ascii')
  above_n = True if age_proof_test == age_is_zero else False

print(f"The user provided valid proof for the age is equal to or greater than {n} test: {above_n}")
```

B.2 HashWires for SD-JWT and MSO

Code examples in Python and descriptions on how to use HashWires for inequality tests for SD-JWT and MSO have been provided by Peter Lee Altmann at the repository "Inequality tests in salted attribute digest based attestations" [i.2].

Annex C: Bibliography

- Ben-or-Goldwasser-Shafi-Kilian-Wigderson: "Multi prover interactive proofs: How to remove intractability assumptions".
- Camenisch-Dubovitskaya-Lehmann: "Concepts and Languages for Privacy-Preserving Attribute-Based Authentication".
- DIF: "Presentation Exchange 2.0.0".
- Dutto-Margaria-Sanna-Vesco: "Toward a Post-Quantum Zero-Knowledge Verifiable Credential System for Self-Sovereign Identity".
- ENISA: "Cybersecurity Certification: Candidate EUCC Scheme V1.1.1".
- ETSI EN 319 102-1: "Electronic Signatures and Infrastructures (ESI); Procedures for Creation and Validation of AdES Digital Signatures; Part 1: Creation and Validation".
- ETSI EN 319 403-1: "Electronic Signatures and Infrastructures (ESI); Trust Service Provider Conformity Assessment; Part 1: Requirements for conformity assessment bodies assessing Trust Service Providers".
- ETSI EN 319 411-2: "Electronic Signatures and Infrastructures (ESI); Policy and security requirements for Trust Service Providers issuing certificates; Part 2: Requirements for trust service providers issuing EU qualified certificates".
- ETSI TR 103 619: "CYBER; Migration strategies and recommendations to Quantum Safe schemes".
- ETSI TS 119 182-1: "Electronic Signatures and Infrastructures (ESI); JAdES digital signatures; Part 1: Building blocks and JAdES baseline signatures".
- FIDO Alliance: "Fast Identity Online v2 (FIDO2)".
- Fuentes-González-Olvera-Veseli: "Assessment of attribute-based credentials for privacy-preserving road traffic services in smart cities".
- IETF RFC 6749: "OAuth 2.0 Authorization Framework".
- ISO/IEC 23220-3: "Cards and security devices for personal identification - Building blocks for identity management via mobile devices - Part 3: Protocols and services for installation and issuing phase".
- OASIS: "PKCS #11 Cryptographic Token Interface Base Specification Version 2.40".
- OpenID Foundation: "OpenID for Verifiable Credential Issuance".
- Rosenberg-White-Garman-Miers: "zk-creds: Flexible Anonymous Credentials from zkSNARKs and Existing Identity Infrastructure".
- W3C®: "JSON-LD 1.1 - A JSON-based Serialization for Linked Data".
- W3C® Working Draft 21 July 2023: "Securing Verifiable Credentials using JOSE and COSE".
- W3C® Working Draft 21 July 2023: "Securing Verifiable Credentials using JOSE and COSE".
- Zhang-Genkin-Katz-Papadopoulos: "vRAM: Faster Verifiable RAM with Program-Independent Preprocessing".

History

Document history		
V1.1.1	August 2023	Publication