On Delegation of Verifiable Presentations from mdoc and BBS Credentials

Andrea Flamini[1] [0000-0002-3872-7251], Andrea Gangemi[1] [0000-0001-9689-8473], Enrico Guglielmino[1] [0009-0004-1394-4761], and Vincenzo Orabona[2]

[1] Department of Mathematical Sciences, Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Torino, Italy

[2] Eustema S.p.A, Via Carlo Mirabello, 7, 00195 Roma, Italy

Abstract. The interest in verifiable credential systems has gained traction as eIDAS 2.0 Regulation has been published. This regulation instructs EU member states to provide their citizens with digital identity wallets (EUDI Wallet) that must store the credentials and enable privacy-preserving presentation of identity information to relying parties. This new digital identity system requires defining new protocols and procedures to perform tasks involving the disclosure of identity information. One of such procedures is the delegation of attestation, as is reported in the EUDI Wallet Reference Implementation Roadmap.

In this work, we address the problem of constructing secure processes for the *delegation of verifiable presentations* derived from both verifiable and anonymous credentials. Our goal is to enable a credential holder (the *delegator*) to securely delegate another party (the *delegatee*) to present a credential on their behalf. We introduce the notion of a *verifiable presentation delegation scheme*, formalizing the core algorithms, namely *delegation issuance*, *delegated presentation*, and *presentation verification*, and defining the relevant security properties that such a scheme should satisfy: the *correctness*, the *unforgeability*, and, when the scheme is built on top of anonymous credentials, even the *unlinkability*. We present two concrete instantiations of delegation schemes: the first is built on top of *mdoc verifiable credentials*, the credential format currently supported by the *EUDI Wallet Architecture and Reference Framework (EUDI ARF)*, while the second is built on top of *BBS anonymous credentials*. Finally, we discuss and analyze the security of our constructions in terms of the security properties we have introduced.

Keywords: Anonymous Credentials · eIDAS 2.0 · Delegation

This work is an extended version of the paper [28] presented at 3rd International Workshop on Trends in Digital Identity (TDI 2025). It includes additional technical details, extended security proofs, and new results that were not part of the original conference publication.

1 Introduction

The eIDAS 2.0 regulation (electronic IDentification, Authentication and Trust Services)[26] aims to regulate electronic identification and trust services in the EU's internal market and to improve cross-border interoperability across Europe by introducing a new digital identity system that will be adopted by each member state. The regulation explicitly requires that this digital identity system provide citizens with the ability to selectively disclose the attributes certified in their credentials and to perform unlinkable authentications, with the goal of protecting their privacy by reducing the amount of information they disclose[1], and to prevent tracking of their activities by colluding relying parties (and issuers).

The solution proposed to achieve these goals is the adoption of verifiable credentials (VCs) stored in a digital wallet called the EUDI Wallet [1]. VCs are the digital analogue of physical credentials, and their security relies on the use of cryptographic techniques. VCs are issued by *issuers* who sign them and deliver them to the credential *holders*. The holder of a VC stores it and can use it to prove specific claims (or statements) about their identity to a *verifier* by creating a verifiable presentation (VP). In general, roles are dynamic and context-dependent, allowing a user to function as a holder, verifier, or issuer based on the specific operations they wish (or are allowed) to perform. For example, an organization acting as a verifier in one transaction can assume the role of an issuer in a subsequent transaction.

Main credential formats. One pivotal decision regarding the design of the EUDI Wallet concerns the choice of types of VCs it should support, enabling the selective disclosure of attributes and the unlinkability of presentations. So far, the community has highlighted two main approaches for the VC format [1,7]. The first approach describes VCs designed exploiting the properties of hiding commitments and standard signatures [3], which we refer to as *mdoc VC*[2], while the second approach describes VCs whose design relies on signature schemes that support Non-Interactive Zero-Knowledge Proofs (NIZKPs) derived from sigma protocols, referred to as *anonymous credentials (AC)*[3]. Although both ACs and mdoc VC support selective disclosure of attributes [30], ACs offer stronger privacy guarantees enabling the generation of predicate proofs (e.g. range proofs) and pseudonyms [37,38,39], but most of all by ensuring that multiple presentations of the same credential are unlinkable even against an adversary who corrupts both the issuer and the verifiers. This property is often referred to as multi-show unlinkability. Clearly multi-show unlinkability is not achieved by mdoc VCs since every presentation always reveals information that identifies the credential used to generate it. To achieve a similar level of privacy for users, mdoc VCs rely on the batch issuance of single-use credentials. This countermeasure protects the holder against corrupted verifiers, but it does not protect the user's privacy if the verifiers collude with the issuer (or if the verifier is the issuer). Additionally, the batch issuance of credentials naturally presents relevant complications for secure deployment. Recently, an approach based on the use of SNARKS has been proposed to provide mdoc VCs with multi-show unlinkability [32].

Delegation of VPs The introduction of this digital identity system, based on the use of verifiable credentials, presents both technical challenges and opportunities to improve authentication security. In this work, we focus on one of the features that the use of verifiable credentials can enable: the ability to delegate the presentation of verifiable credentials to a different holder without involving any trusted third party, in a cryptographically verifiable manner. The relevance of this feature is attested by its inclusion in the EUDI Wallet Reference Implementation Roadmap [2].

A credential holder (the delegator), using its VC or AC, can create a delegation that instructs another holder (the delegatee) to act on its behalf to execute a *specific and predetermined* operation while interacting with a verifier. VP delegation is developed as a

[1] In accordance with the privacy principles required by the GDPR [25]
[2] This name refers to the mobile driving license described in the standard ISO/IEC 18013-5[35].
[3] In particular the focus is on ACs designed following the framework of Camenisch and Lysyanskaya [15].

mechanism allowing one to cover use cases such as: someone authorizes a family member to pick up a prescription from the pharmacy, an older adult authorizes their adult child to represent them in a public online service, or a person delegates an intermediary to perform financial operations on their behalf.

For simplicity, in this work, we consider a system in which every user is provided a "main credential", such as an ID card, by a trusted issuer that every holder can use to delegate to other holders. Our definition of a VP delegation scheme builds on top of such a VC or AC scheme. In particular, we describe a delegation mechanism that allows a delegator to generate delegations that must specify:

- the *delegatee identifier*, a statement that the identity (or the VC/AC) of the delegatee must satisfy. For example, the delegatee identifier could be "the delegatee is over 18", then any over 18 holder would be allowed to present such delegation, or more specific statements such as "the delegatee name is *Name* and their surname is *Surname*", in such a case only a holder named *Name Surname* would be allowed to present such delegation. This identifier is sent by the delegatee to the delegator before the latter produces the delegation;
- the *scope* for which the delegation is being created. This might include a period of validity, an identifier of the verifier to which it must be presented, and an identifier of the operation that the delegatee must perform;
- the *delegator payload*, a statement about the delegator's identity that contains (at least) the information the verifier would request from any holder to perform the operation specified in the scope;
- a *proof* that the delegator identity (or VC/AC) satisfies the delegator payload.

The delegatee, holding the delegation, interacts with the verifier to present the delegation and executes the operations specified in the scope. The delegatee proves to be an authorized delegatee by showing, using its own credential, that it satisfies the delegatee identifier information specified in the delegation. Finally, the verifier checks that the delegation and the presentation of the delegation are valid and accepts (or rejects) the delegated presentation, allowing the delegatee to perform (or preventing the delegatee from performing) the operation specified in the scope.

Our Contribution. Unlike the delegation of issuance of anonymous credentials, a topic widely studied in the cryptographic literature, for instance [20,8,9,13,22,33], to the best of our knowledge, delegation of VPs has not been formalized in the existing literature. However, in order to ensure the secure deployment of the delegation of VP functionality, it is necessary to formally define both its syntax and the security properties that it must satisfy to enable a security analysis of its instantiations. For this reason, we fill this gap and provide a formal definition of VP delegation scheme as an extension that is built on top of a verifiable credential scheme.

We define the syntax of a VP delegation scheme and we identify the security properties that it must satisfy, namely the *correctness*, that guarantees that legitimate delegators can delegate third parties to perform a presentation on their behalf; the *unforgeability of delegated presentations*, that captures that an adversary should not be able to present a delegation for which it is not a legitimate delegatee, and should not be able to generate a delegation for a delegator payload that is not certified by the credentials it controls; finally the *unlinkability* that captures the ability to generate delegations, and delegated presentations that can not be linked to a specific credential issued by the issuer. Then, we describe a VP delegation scheme that can be applied to mdoc VCs and another one that can be applied to ACs: for the sake of concreteness, the VP delegation scheme is instantiated with BBS anonymous credentials [14,43,11], but it can be easily generalized to any AC. We prove the security of our VP delegation schemes according to our security notion. We prove the unforgeability of both schemes (in particular, we observe that the proof of the scheme based on anonymous credentials is remarkably simpler than the one based on mdoc VCs) and we prove the unlinkability of the scheme based on AC. To conclude, we briefly discuss the compatibility of our scheme in the context of the EUDI and EBSI frameworks.

Structure of the paper. In Section 2, we provide the necessary background by introducing some preliminary concepts. Section 3 provides a detailed overview of some relevant verifiable credential schemes — specifically the mdoc VC scheme and the BBS AC scheme — for which we will explicitly define a VP delegation scheme in Section 4. The security notions of VP delegation scheme is introduced in Section 4 formalizes the security definitions that a delegation scheme must satisfy; specifically, for verifiable credentials, we define correctness and unforgeability, while for anonymous credentials we also require unlinkability. Section 5 provides concrete instantiations of our VP delegation framework over the credential systems introduced in Section 3, resulting in two constructions: one based on mdoc VC and the other based on anonymous credentials derived from BBS signatures. Section 6 contains the security analysis of both constructions, including formal proofs of correctness, unforgeability, and unlinkability as defined in Section 4. Finally, in Section 7, we discuss how our protocol can be integrated into real-world ecosystems, such as EBSI and EUDI.

2 Preliminaries

In this section, we provide the notation and the necessary background, including the definition of *Digital signatures*, *Sigma protocols*, *Non-Interactive Zero-Knowledge Proofs* and *Commitment schemes.*

2.1 Notation

Let λ denote the security parameter. The issuer's private and public keys are denoted as $\mathsf{sk}_{\mathsf{Iss}}$ and $\mathsf{pk}_{\mathsf{Iss}}$, respectively. The set of messages or attributes is denoted by $\mathbf{a} = (a_i)_{i\in[l]}$, where $[l]$ represents the set $\{1, \ldots, l\}$.

When presenting a VC/AC supporting selective disclosure, a subset of attributes, whose indices are denoted by Hid, can be hidden, while the set of indices of the disclosed attributes is $\mathsf{Rev} = [l] \setminus \mathsf{Hid}$. We denote by stmt the set of revealed attributes, and by $\mathcal{S}$ the set of possible statements.

The *delegator*, *delegatee*, and *verifier* are indicated by D, Δ, and V, respectively. The delegatee identifier is indicated by Δ_{ID}, the delegator payload is indicated as DP, while the delegation scope is referred to as scope.

For simplicity, we assume each party is provided with a single VC/AC. The delegator's credential is represented as $\mathsf{cred}_{\mathsf{D}}$, and the delegatee's credential is denoted as cred_{Δ}.

In the security definitions, we will define the following tables: a credential table CT, a presentation table PT, and a delegation table DT. Finally, we consider a function $f : \mathbb{N} \to \mathbb{R}$ to be *negligible* if, for every $c > 0$, there exists an integer $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$, $f(n) = O(1/n^c)$. This notation will be used throughout the paper to formalize the schemes and algorithms presented.

2.2 Digital signatures

Digital signatures are cryptographic primitives that ensure the authenticity and integrity of messages. They allow a sender to produce a signature on a message such that any verifier, given the sender's public key, can check its validity (see [12] for more details).

Definition 1 (Digital Signature Scheme). *A* digital signature scheme $\mathcal{DS}$ *consists of four efficient algorithms* $= (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vf})$ *such that:*

– Setup $\mathsf{pp} \leftarrow \mathsf{Setup}(\lambda)$*: this algorithm takes as input the security parameter* λ *and outputs the public parameters* pp *used by the scheme.*
– Key Generation $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$*: this algorithm takes as input the public parameters* pp *and outputs a pair* $(\mathsf{pk}, \mathsf{sk})$*, where* sk *is the secret* signing key*, and* pk *is the public* verification key.
– Signing $\sigma \leftarrow \mathsf{Sign}(m, \mathsf{sk})$*: A probabilistic algorithm that, given a secret key* sk *and a message* m*, outputs a* signature σ*.*
– Verification $(\mathsf{accept}/\mathsf{reject}) \leftarrow \mathsf{Vf}(\sigma, m, \mathsf{pk})$*: A deterministic algorithm that, given a public key* pk*, a message* m*, and a signature* σ*, outputs either* accept *or* reject*.*

A digital signature scheme must satisfy correctness *and* unforgeability.

– Correctness*: for all key pairs* $(\mathsf{sk}, \mathsf{pk})$ *output by* KeyGen*, and for all messages* m*, any signature* σ *generated by* Sign *must be accepted by* Vf*. Formally,*

$$\Pr[\mathsf{Vf}(\sigma, m, \mathsf{pk}) = \mathsf{accept}] = 1.$$

– Unforgeability *(informal): if an adversary is able to obtain valid pairs* $\{(m_1, \sigma_1), \ldots, (m_k, \sigma_k)\}$*, they still cannot output a valid message-signature pair* (m, σ) *for some new message* m*. Digital signatures that satisfy this property are said to be* unforgeable under a chosen message attack *(UF-CMA security).*
– Strong Unforgeability *(informal): if an adversary is able to obtain valid pairs* $\{(m_1, \sigma_1), \ldots, (m_k, \sigma_k)\}$*, they still cannot output a valid message-signature pair* $(m, \sigma) \neq (m_i, \sigma_i),\ \forall i \in [k]$*. Digital signatures that satisfy this property are said to be* strongly unforgeable under a chosen message attack *(SUF-CMA security).*

The formal definitions of unforgeability *and* strong unforgeability *for digital signature schemes are given in Appendix .1.*

2.3 Sigma Protocols

Sigma protocols are a class of interactive proofs that allow a prover to convince a verifier of the validity of a statement while satisfying specific security properties.

Definition 2 (Sigma protocol). *Let* $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{Y}$ *be a* NP *relation. A* Sigma protocol Π *for* $\mathcal{R}$ *is an interactive proof system consisting of two algorithms:*

– *The* prover P*, which is a probabilistic algorithm that takes as input a pair* $(x, y) \in \mathcal{R}$*, where* x *is the witness and* y *is the associated statement.*
– *The* verifier V*, which is a probabilistic algorithm that takes as input the statement* $y \in \mathcal{Y}$ *and outputs either* accept *or* reject*.*

The interaction between P *and* V *follows a three-step protocol:*

1. *The prover generates a* commitment t *and sends it to the verifier.*
2. *The verifier chooses a random* challenge c *from a finite challenge space* $\mathcal{C}$ *and sends it to the prover.*
3. *The prover computes a* response z *based on* c*, and sends it back to the verifier.*
4. *The verifier uses* transcript (t, c, z) *and the statement* y *to calculate a deterministic decision:* accept *or* reject*.*

Note that the verifier's behavior is deterministic, except for the random choice of the challenge c*.*

A sigma protocol must satisfy three security properties: correctness*,* special soundness*, and* honest-verifier zero-knowledge *(see [12] for formal definitions).*

2.4 Non-Interactive Zero-Knowledge Proofs

Definition 3 (Non Interactive Zero-Knowledge Proof). *Let* $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{Y}$ *be an* NP *relation. A* Non Interactive Zero-Knowledge Proof *(NIZKP) for* $\mathcal{R}$ *consists of two algorithms* (P, V)*, where* P *has in input a witness-statement pair* $(x, y) \in \mathcal{R}$*, while* V *only has in input the statement* y*. The prover algorithm outputs a proof* $\pi \leftarrow \mathsf{P}(x, y)$ *which is sent to the verifier who can verify it running* $\mathsf{reject}/\mathsf{accept} \leftarrow \mathsf{V}(y, \pi)$*.*

The algorithm must satisfy the following properties:

1. Completeness*: For* $(x, y) \in \mathcal{R}$*,* $\mathsf{V}(\mathsf{P}(x, y), y) = 1$*;*
2. Zero-Knowledge*: For any PPT distinguisher* $\mathcal{D}$*, there exists a PPT simulator* Sim *such that, for any statement* y *chosen by* $\mathcal{D}$*, the following two distributions are computationally indistinguishable:*
$$\{\pi \leftarrow \mathsf{P}(x, y) : (x, y) \in \mathcal{R}\} \quad \approx_c \quad \{\pi' \leftarrow \mathsf{Sim}(y)\}.$$
In other words, even without knowing the witness x*, the simulator can generate a proof* π' *for* y *that is indistinguishable from a proof* π *honestly generated by a prover who knows* x*;*
3. Knowledge Soundness*: There exists a probabilistic polynomial-time (PPT) algorithm* Ext*, called the* extractor*, such that for any PPT adversary* $\mathcal{A}$*, if* $\mathcal{A}$ *can produce a valid proof* π *for some statement* y *with non-negligible probability, then given* y *and rewindable oracle access to* $\mathcal{A}$*, the extractor* Ext *can extract a witness* x *such that* $(x, y) \in \mathcal{R}$ *with non-negligible probability.*

The Fiat-Shamir transform can be used to convert a sigma protocol into a non-interactive zero-knowledge proof.

Definition 4 (Fiat-Shamir NIZKP). *Let* $\Pi = (\mathsf{P}, \mathsf{V})$ *be a sigma protocol for a* NP *relation* $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{Y}$*. Assume that the transcripts* (t, c, z) *belong to* $\mathcal{T} \times \mathcal{C} \times \mathcal{Z}$*. Let* $H : \mathcal{Y} \times \mathcal{T} \to \mathcal{C}$ *be a cryptographic hash function. The* Fiat-Shamir transform *converts* Π *into a non-interactive zero-knowledge proof as follows:*

1. *Given an input* $(x, y) \in \mathcal{R}$*, the prover generates a commitment* $t \in \mathcal{T}$*;*
2. *The prover computes the challenge as* $c = H(y, t)$*, then they compute a response* $z \in \mathcal{Z}$*; finally, the prover sends* $(t, z) \in \mathcal{T} \times \mathcal{Z}$ *to the verifier;*
3. *Given* $(y, (t, z)) \in \mathcal{Y} \times (\mathcal{T} \times \mathcal{Z})$*, the verifier computes* $c = H(y, t)$ *and checks whether* (t, c, z) *is an accepting transcript for* y*.*

In certain settings, the challenge c *can be sent to* V *instead of the commitment* t *(or part of it), provided that the protocol admits a deterministic reconstruction of the commitment, thus reducing the communication cost.*

2.5 Commitment schemes

A *commitment scheme* allows a party (the *committer*) to commit to a message m by generating and sending a commitment value com, while satisfying the following properties:

- *Hiding (informal)*: The commitment com does not reveal any information about the message m.
- *Binding (informal)*: It is computationally infeasible for the committer to generate a commitment com that can be opened to two distinct messages m and m'.

For formal definitions of the hiding and binding security experiments, see [36].

Definition 5 (Commitment Scheme). *A* (non-interactive) commitment scheme *consists of three algorithms,* Gen*,* Com *and* Open*, and operates as follows:*

1. *The key generation algorithm* $\mathsf{Gen}(\lambda)$ *takes as input the security parameter* λ *in unary and outputs the public parameters* pp.
2. *To commit to a message* $m \in \{0,1\}^\lambda$*, the sender selects a uniformly random value* r*, computes the commitment* $\mathsf{com} = \mathsf{Com}(\mathsf{pp}, m; r)$*, and sends* com *to the receiver.*
3. *To open the commitment, the sender reveals* (m, r) *to the receiver.*
4. *The receiver runs the algorithm* $\mathsf{Open}(\mathsf{pp}, m, \mathsf{com}; r)$ *and outputs* accept *or* reject.

A secure commitment scheme can be easily constructed using a cryptographic hash function H. The commitment based on a hash function H is defined as follows: to commit to a message m, sample a random $\mathsf{salt} \xleftarrow{\$} \{0,1\}^\lambda$ and compute $\mathsf{com} = H(m||\mathsf{salt})$. To open a commitment com, it is necessary to reveal the message-salt pair (m, salt), and the verifier can check if $\mathsf{com} = H(m||\mathsf{salt})$ (equivalently, $\mathsf{Open}(m, \mathsf{com}; \mathsf{salt}) = \mathsf{accept}$). This commitment scheme can be proved to be hiding and binding.

3 Verifiable Credential schemes

In this section, we introduce the verifiable credential schemes that we analyze in this work and for which we provide an explicit description of a VP delegation scheme.

A *verifiable credential scheme* is defined by a set of algorithms and protocols that establish secure processes to issue, present and verify digital credentials [30].

Definition 6 (Verifiable Credential Scheme). *A* verifiable credential scheme *is defined by the following algorithms:*

- Issuer Setup *(*$\{\mathsf{pp}, (\mathsf{sk_{Iss}}, \mathsf{pk_{Iss}})\} \xleftarrow{\$} \mathsf{IssuerSetup}(\lambda)$*): this algorithm is executed by the issuer, and it generates the public parameters* pp *for the verifiable credential scheme together with the issuer key pair* $(\mathsf{sk_{Iss}}, \mathsf{pk_{Iss}})$.
- Credential Issuance *(*$\mathsf{cred} \xleftarrow{\$} \mathsf{CredIssuance}(\{a_i\}_{i\in[l]}, \mathsf{sk_{Iss}})$*): The issuer executes this algorithm to generate a verifiable credential* cred *for the attributes* $\{a_i\}_{i\in[l]}$.
- Credential Presentation *(*$\mathsf{pres} \xleftarrow{\$} \mathsf{CredPresentation}(\mathsf{cred}, \mathsf{stmt})$*): This algorithm is executed by the holder to generate a proof (the presentation* pres*) that it possesses a credential* cred *that satisfies a statement* stmt[4].

[4] The presentation can allow the holder to selectively disclose the attributes of the credential, revealing only the attributes in stmt.

– Presentation Verification *($\{0,1\} \overset{\$}{\leftarrow} \mathsf{PresVer}(\mathsf{pres}, \mathsf{stmt}, \mathsf{pk}_{\mathsf{Iss}})$): The verifier executes this algorithm to verify the validity of the presentation provided by the holder, ensuring that it satisfies* stmt.

A VC scheme must satisfy two fundamental security properties. The first one is *correctness*, which ensures that a verification protocol succeeds if and only if the prover presents a valid credential that was previously issued by a recognized issuer through the designated issuance protocol. The second is *unforgeability*, meaning that no adversary should be able to generate a valid presentation for a statement which is not satisfied by any of the credentials it was previously issued.

In this work, we focus on the class of VCs schemes that are currently adopted in the EUDI Wallet Architecture Reference Framework (ARF) [1, Annex 2, Topic 12], or are considered for possible future adoption. According to the EUDI ARF, the supported VC schemes must have one of the following formats: mdoc, described in [35,3], SD-JWT, described in [27], or W3C VCDM, described in [42]. These different verifiable credential formats are based on the same cryptographic mechanism that we abstract in this section, and, for simplicity, we will refer to it as *mdoc VC*, as previously mentioned. The VC scheme considered for possible future adoption by the EUDI ARF is the BBS anonymous credentials [11,14], as also advised in [7], which has recently attracted significant attention in the cryptographic literature, with papers that analyse its security [43,19,18], that build distributed variants from it [24,34,40,29], and papers that presenting efforts to make it eIDAS compliant [23,17,31].

We now describe these two verifiable credential schemes. In the rest of this work, the public parameters pp are omitted from the input of most described algorithms for notational simplicity.

3.1 mdoc VC scheme

The cryptographic primitives used to design mdoc VCs are (1) hiding commitments based on hash functions and (2) digital signature schemes $\mathcal{DS} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vf})$. The digital signature scheme $\mathcal{DS}$ can be any SUF-CMA secure digital signature scheme.

Definition 7 (mdoc VC scheme). *Let $\mathcal{DS} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vf})$ be a digital signature scheme and H a cryptographic hash function. The* mdoc VC scheme *can be described as follows:*

– Issuer setup: $\{\mathsf{pp}, (\mathsf{sk}_{\mathsf{Iss}}, \mathsf{pk}_{\mathsf{Iss}})\} \overset{\$}{\leftarrow} \mathsf{IssuerSetup}(\lambda)$.
The issuer executes the $\mathsf{IssuerSetup}$ *algorithm which consists in executing the* Setup *and* KeyGen *algorithms of the underlying digital signature scheme $\mathcal{DS}$. This algorithm generates:*
  - *the public parameters:* $\mathsf{pp} \overset{\$}{\leftarrow} \mathsf{Setup}(\lambda)$*;*
  - *the issuer key pair:* $(\mathsf{sk}_{\mathsf{Iss}}, \mathsf{pk}_{\mathsf{Iss}}) \overset{\$}{\leftarrow} \mathsf{KeyGen}(\mathsf{pp})$.

– Credential issuance: $\mathsf{cred} \overset{\$}{\leftarrow} \mathsf{CredIssuance}(\{a_i\}_{i\in[l]}, \mathsf{sk}_{\mathsf{Iss}})$.
The issuer executes the following operations:
  - *sample uniformly at random salts* $\mathsf{salt}_1, \ldots, \mathsf{salt}_l \overset{\$}{\leftarrow} \{0,1\}^\lambda$*;*
  - *compute* $\mathsf{com}_i \leftarrow H(a_i || \mathsf{salt}_i)\, \forall i \in [l]$*;*

  - *generate* $(\mathsf{sk}_{\mathsf{cred}}, \mathsf{pk}_{\mathsf{cred}}) \overset{\$}{\leftarrow} \mathsf{KeyGen}(\mathsf{pp})$;
  - *sign the commitments and the public key of the credential* $(\{\mathsf{com}_i\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}})$ *computing* $\sigma \overset{\$}{\leftarrow} \mathsf{Sign}((\{\mathsf{com}_i\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}}), \mathsf{sk}_{\mathsf{Iss}})$;
  - *set* $\mathsf{cred} \leftarrow ((\sigma, \{\mathsf{com}_i\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}}), \{a_i\}_{i\in[l]}, \{\mathsf{salt}_i\}_{i\in[l]}, \mathsf{sk}_{\mathsf{cred}})$[5].

– Credential presentation*:* $\mathsf{pres} \overset{\$}{\leftarrow} \mathsf{CredPresentation}(\mathsf{cred}, \mathsf{stmt}, \mathsf{nonce})$.
The statement $\mathsf{stmt} = \{a_i\}_{i\in\mathsf{Rev}}$ *is given by the set of attributes that the holder wants to reveal* $\mathsf{Rev} \subseteq [l]$*, and the nonce* nonce *is sent by the verifier to the holder to guarantee the freshness of the presentation. The holder performs the following operations:*
  - *compute* $\mathsf{pres}' \leftarrow ((\sigma, \{\mathsf{com}_i\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}}), \{\mathsf{salt}_i\}_{i\in\mathsf{Rev}}, \{a_i\}_{i\in\mathsf{Rev}}, \mathsf{nonce})$;
  - *sign* pres' *computing* $\sigma' \overset{\$}{\leftarrow} \mathsf{Sign}(\mathsf{pres}', \mathsf{sk}_{\mathsf{cred}})$;
  - *set* $\mathsf{pres} \leftarrow (\mathsf{pres}', \sigma')$.

– Presentation Verification*:* $\{0,1\} \overset{\$}{\leftarrow} \mathsf{PresVer}(\mathsf{pres}, \mathsf{stmt}, \mathsf{pk}_{\mathsf{Iss}})$.
The verifier performs the following operations:
  - *parse* $\mathsf{pres} \rightarrow (\mathsf{pres}', \sigma')$ *and then* $\mathsf{pres}' \rightarrow ((\sigma, \{\mathsf{com}_i\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}}), \{\mathsf{salt}_i\}_{i\in\mathsf{Rev}}, \{a_i\}_{i\in\mathsf{Rev}}, \mathsf{nonce})$;
  - *verify the signature of the issuer:* $1 \leftarrow \mathsf{Vf}(\sigma, (\{\mathsf{com}_i\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}}), \mathsf{pk}_{\mathsf{Iss}})$;
  - *check that* $\mathsf{com}_i = H(a_i||\mathsf{salt}_i), \forall i \in \mathsf{Rev}$;
  - *verify the signature of the holder:* $1 \leftarrow \mathsf{Vf}(\sigma', \mathsf{pres}', \mathsf{pk}_{\mathsf{cred}})$.

If the previous checks are satisfied, the verifier accepts and outputs 1, otherwise it outputs 0.

Privacy Concerns for mdoc VCs. This kind of VC offer limited privacy protection because each presentation of the same credential can be linked to the original credential (e.g. the signature of the issuer, or the public key of the credential must be always revealed). This linkability implies that repeated use of the same credential may allow verifiers to track and correlate different interactions, thereby undermining user privacy. For this reason the EUDI ARF currently instructs the member states to implement batch issuance of these VCs that must be considered as single-use credentials. However, even if these credentials are used only once, if the verifier colludes with the issuer, or it is the same issuer who generated the VC, then the VC presentation can be easily linked to its issuance.

3.2 BBS AC scheme

Anonymous credentials [21,15] are the tool recommended for addressing the linkability problem of mdoc VCs. The privacy-preserving property that guarantees that multiple presentations of the same credential can not be linked is referred to as *unlinkability*, and in the case of BBS anonymous credentials [11,14,43] (as for all the schemes that follow the framework of Camenisch and Lysyanskaya [15] like [41,16,14]), the unlinkability is achieved through the use of NIZKP that allow the prover to demonstrate possession of a valid credential while revealing no information beyond the disclosed statement.

In Appendix .2, we provide a general description of the algorithms for an anonymous credential scheme constructed according to the framework of Camenisch and Lysyanskaya

[5] To ensure a secure binding between a credential and the device storing it, the secret key of the credential ($\mathsf{sk}_{\mathsf{cred}}$) is stored within a hardware security module (HSM) or a trusted execution environment (TEE) embedded in the device.

[15]. Here, we explicitly describe the algorithms for the BBS anonymous credential scheme as described in [43].

Let $\mathbb{G}_1 = \langle g_1 \rangle$, $\mathbb{G}_2 = \langle g_2 \rangle$, and $\mathbb{G}_T$ be groups of prime order p. A pairing [6] $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a map that satisfies the following properties:

– Bilinearity: $\mathbf{e}(g_1^x, g_2^y) = \mathbf{e}(g_1, g_2)^{xy} \quad \forall x, y \in \mathbb{F}_q^*$.
– Non-degenerate: $\mathbf{e}(g_1, g_2) \neq 1$.
– Efficient computability: the pairing $\mathbf{e}$ can be computed in polynomial time with respect to the bit length of the security parameter λ.

The BBS signature scheme is defined as follows.

Definition 8 (BBS signature scheme). *The signature scheme is defined as follows:*

– $\mathsf{pp} \overset{\$}{\leftarrow} \mathsf{Setup}(\lambda)$*: Outputs a bilinear group given by* $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$*, and the pairing* $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ *as defined above, and random* $(g_1, g_2, \mathbf{h} = (h_i)_{i \in [l]}) \overset{\$}{\leftarrow} \mathbb{G}_1^{l+2}$.
– $(\mathsf{sk}, \mathsf{pk}) \overset{\$}{\leftarrow} \mathsf{KeyGen}(\mathsf{pp})$*: Sample uniformly at random* $x \overset{\$}{\leftarrow} \mathbb{Z}_p$*, compute* $X = g_2^x \in \mathbb{G}_2$*, and set* $\mathsf{sk} = x$ *and* $\mathsf{pk} = X$.
– $(A, e) \overset{\$}{\leftarrow} \mathsf{Sign}(\mathbf{a}, \mathsf{sk} = x)$*: On input messages* $\mathbf{a} = (a_i)_{i \in [l]}$*, sample uniformly at random* $e \overset{\$}{\leftarrow} \mathbb{Z}_p$ *and compute*

$$A = \left(g_1 \prod_{i=1}^{l} \mathbf{h}[i]^{\mathbf{a}[i]} \right)^{\frac{1}{x+e}}.$$

Output the signature (A, e).
– $0/1 \leftarrow \mathsf{Vf}((A, e), \mathbf{a}, \mathsf{pk} = X)$*: compute* $C(\mathbf{a}) = g_1 \prod_{i=1}^{l} \mathbf{h}[i]^{\mathbf{a}[i]}$ *and check if*

$$\mathbf{e}(A, X g_2^e) = \mathbf{e}(C(\mathbf{a}), g_2).$$

Return 1 if the check is correct, and 0 otherwise.

On top of this digital signature scheme it is possible to define the following anonymous credential scheme.

Definition 9 (BBS AC scheme).
The BBS anonymous credential scheme is defined by the following algorithms.

– Issuer setup: $\{\mathsf{pp}, (\mathsf{sk}_{\mathsf{Iss}}, \mathsf{pk}_{\mathsf{Iss}})\} \overset{\$}{\leftarrow} \mathsf{IssuerSetup}(\lambda)$.
The issuer executes the $\mathsf{IssuerSetup}$ *algorithm which consists in generating the public parameters* pp *for the BBS signature (Def. 8 scheme along with the key pair of the issuer* $(\mathsf{sk}_{\mathsf{Iss}}, \mathsf{pk}_{\mathsf{Iss}})$.

$$\{\mathsf{pp}, (\mathsf{sk}_{\mathsf{Iss}}, \mathsf{pk}_{\mathsf{Iss}})\} \overset{\$}{\leftarrow} \mathsf{IssuerSetup}(\lambda),$$

where $\mathsf{pp} = (g_1, (h_i)_{i=1}^{l}, g_2, \mathsf{pk}_{\mathsf{Iss}}, \mathbf{e})$*,* $\mathsf{sk}_{\mathsf{Iss}} = x \overset{\$}{\leftarrow} \mathbb{Z}_p$*, and* $\mathsf{pk}_{\mathsf{Iss}} = X_2 = g_2^x$.

[6] The most efficient implementations of the BBS signature are based on the curve BLS 12-381 [6] which defines a pairing such that there is not an efficient isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$, and where computations in $\mathbb{G}_1$ are more efficient than the computations in $\mathbb{G}_2$.

– Credential issuance: $\mathsf{cred} \overset{\$}{\leftarrow} \mathsf{CredIssuance}(\{a_i\}_{i\in[l]}, \mathsf{sk}_{\mathsf{Iss}})$.
  The issuer executes this algorithm to generate a BBS anonymous credential. It takes as input the attributes $(a_1, \ldots, a_l)$ *and the issuer's private key* $\mathsf{sk}_{\mathsf{Iss}}$*, samples* $e \overset{\$}{\leftarrow} \mathbb{Z}_p$*, computes*

$$A = \left(g_1 \prod_{i\in[l]} h_i^{a_i}\right)^{\frac{1}{x+e}},$$

  and outputs a credential cred*:*

$$\mathsf{cred} = (\sigma, \{a_i\}_{i\in[l]}) = ((A, e), \{a_i\}_{i\in[l]}) \overset{\$}{\leftarrow} \mathsf{CredIssuance}(\{a_i\}_{i\in[l]}, \mathsf{sk}_{\mathsf{Iss}}).$$

  This credential cred *includes the attributes* $(a_1, \ldots, a_l)$ *and its signature* (A, e) *obtained using the BBS signature scheme with secret key* $\mathsf{sk}_{\mathsf{Iss}} = x$.
– Credential presentation: $\mathsf{pres} \overset{\$}{\leftarrow} \mathsf{CredPresentation}(\mathsf{cred}, \mathsf{stmt}, \mathsf{nonce})$.
  The statement $\mathsf{stmt} = \{a_i\}_{i\in\mathsf{Rev}}$ *is given by the set of attributes that the holder wants to reveal* $\mathsf{Rev} \subseteq [l]$*, and the* nonce *is sent by the verifier to the holder to guarantee the freshness of the presentation. To produce a presentation, the holder executes the* $\mathsf{CredPresentation}$ *algorithm described in Figure 1:*

$$\mathsf{pres} = (\bar{A}, \bar{B}, c, s, t, (u_i)_{i\in\mathsf{Hid}}) \overset{\$}{\leftarrow} \mathsf{CredPresentation}(\mathsf{cred}, \mathsf{stmt}, \mathsf{nonce}),$$

– Presentation Verification: $\{0,1\} \overset{\$}{\leftarrow} \mathsf{PresVer}(\mathsf{pres}, \mathsf{stmt}, \mathsf{pk}_{\mathsf{Iss}})$.
  The verifier executes the the $\mathsf{PresVer}$ *algorithm described in Figure 1: if the checks are satisfied, the verifier accepts and outputs 1, otherwise it outputs 0.*

The $\mathsf{CredPresentation}$ *and* $\mathsf{PresVer}$ *algorithms are illustrated in Figure 1, which provides an overview of the non-interactive zero-knowledge proof (NIZKP) construction for BBS credentials.*

In [43], it was proved that this NIZKP is secure since it is derived from a sigma protocol to which is applied the Fiat-Shamir transform. In [43] the authors show that the sigma protocol satisfies *correctness*, *special soundness*, *honest-verifier zero-knowledge*. Moreover, it is easy to see that the sigma protocol has a large challenge space and high first message min-entropy; therefore, the derived NIZKP is secure [4].

4 VP Delegation Scheme: Definition and Security Notions

In this section, we define the notion of a *VP delegation scheme*, which is built on top of a VC scheme, as defined in Definition 6. A VP delegation scheme must define the following algorithms: (1) a delegation issuance algorithm, to allow the delegator D to create a delegation del, (2) a delegation verification algorithm to check the validity of a delegation, (3) a delegation presentation algorithm, to allow the delegatee Δ to present del to a verifier, generating a presentation pres, and (4) to verify the delegated presentation pres.

$$
\begin{array}{lcl}
\textbf{Prover } P(A, e, \mathbf{a}, \mathsf{pp}) & & \textbf{Verifier } V(\mathsf{pp}) \\
r \leftarrow \mathbb{Z}_p^* & & \\
\bar{A} \leftarrow A^r, \quad \bar{B} \leftarrow C(\mathbf{a})^r \bar{A}^{-e} & & \\
\alpha, \beta \leftarrow \mathbb{Z}_p, \quad \delta_i \leftarrow \mathbb{Z}_p \quad \forall\, i \in \mathsf{Hid} & & \\
U \leftarrow \bar{A}^{\alpha} \bar{B}^{\beta} \prod_{i \in \mathsf{Hid}} \mathbf{h}[i]^{\delta_i} & & \\
c \leftarrow H(\bar{A}, \bar{B}, U) & & \\
s \leftarrow \alpha + r^{-1} \cdot e \cdot c & & \\
n \leftarrow \beta + r^{-1} \cdot c & & \\
u_i \leftarrow \delta_i - \mathbf{a}[i]\, c \quad \forall\, i \in \mathsf{Hid} & \xrightarrow{\bar{A}, \bar{B}, c, s, n, (u_i)_{i \in \mathsf{Hid}}} & \\
 & & \tilde{U} \leftarrow \bar{A}^s\, \bar{B}^n \prod_{i \in \mathsf{Hid}} \mathbf{h}[i]^{u_i}\, C_{\mathsf{Rev}}(\mathbf{a})^{-c} \\
 & & c \overset{?}{=} H(\bar{A}, \bar{B}, \tilde{U}) \\
 & & \mathbf{e}(\bar{A}, X_2) \overset{?}{=} \mathbf{e}(\bar{B}, g_2)
\end{array}
$$

Fig. 1. NIZKP for BBS credentials. The value pp represents a set of public parameters known both to P and V.

VP Delegation Scheme. We formally define the input-output specifications of each algorithm that constitutes a VP delegation scheme.

Definition 10 (VP delegation scheme). *A* VP delegation scheme $\mathcal{VPDS}$*, built on top of a VC or AC scheme (*$\mathsf{IssuerSetup}, \mathsf{CredIssuance}, \mathsf{CredPresentation}, \mathsf{PresVer}$*), is defined by the the following algorithms:*

– Delegation issuance*:* $\underbrace{(\Delta_{\mathsf{ID}}, \mathsf{scope}, \mathsf{DP}, \pi_{\mathsf{DP}})}_{\mathsf{del}} \overset{\$}{\leftarrow} \mathsf{DelegIssuance}(\mathsf{cred_D}, \mathsf{DP}, \mathsf{scope}, \Delta_{\mathsf{ID}})$*, where:*

- Δ_{ID} *is the* delegatee identity *which is a statement that must be satisfied by the delegatee presenting* del *(and its credential).* Δ_{ID} *represents a set of attributes with the associated indices that the delegatee must reveal when presenting the delegation. With a slight abuse of notation we can say that, being* $\{a_i\}_{i \in [l]}$ *the attributes in the credential,* $\Delta_{\mathsf{ID}} = \{a_i\}_{i \in \Delta_{\mathsf{ID}}}$*;*
- scope *is the* delegation scope*, describing the operation that can be performed by the delegatee interacting with specified verifiers* [7]*;*
- DP *is the* delegator payload *containing the attributes that the delegator must disclose about its identity (e.g. it is entitled to withdraw a specific drug) when it creates its delegation. As for* Δ_{ID}*, we write* $\mathsf{DP} = \{a_i\}_{i \in \mathsf{DP}}$*;*
- π_{DP} *is a proof that the holder knows a credential* $\mathsf{cred_D}$ *for the claims in* DP*. The proof must be bound to* scope *and* Δ_{ID}*.*

See Section 1 for a better intuition about the semantic meaning of these fields.

[7] Note that these information must be encoded using a dictionary or schema which subsequently allows verifiers of delegated presentations to verify it. A more detailed analysis of this aspect is out of scope and will be treated in a future work.

- Delegation verification*:* $\{0,1\} \xleftarrow{\$} \mathsf{DelegVer}(\mathsf{del}, \mathsf{pk}_{\mathsf{Iss}})$.
  This algorithm is executed by a delegatee (or by a verifier) to verify the validity of the delegation.
- Delegated presentation*:* $\underbrace{(\mathsf{del}, \pi_{\mathsf{del}})}_{\mathsf{pres}} \xleftarrow{\$} \mathsf{DelegPres}(\mathsf{del}, \mathsf{cred}_{\Delta}, \mathsf{nonce})$.
  The verifier sends to the delegatee a random nonce *which is used to guarantee the freshness of the delegated presentation. After parsing* del *as* $(\Delta_{\mathsf{ID}}, \mathsf{scope}, \mathsf{DP}, \pi_{\mathsf{DP}})$*, the delegatee computes* π_{del} *which is a proof of knowledge of a credential* cred_{Δ} *satisfying the delegatee identity* Δ_{ID} *included in* del*.*
- Delegated presentation verification*:* $\{0,1\} \xleftarrow{\$} \mathsf{DelegPresVer}(\mathsf{pres}, \mathsf{pk}_{\mathsf{Iss}})$.
  This is the verification algorithm executed by the verifier. Parse pres *as* $(\mathsf{del}, \pi_{\mathsf{del}})$*. Upon receiving the proof* π_{del} *and the delegation* del*, the verifier checks that*
  - $1 \leftarrow \mathsf{DelegVer}(\mathsf{del}, \mathsf{pk}_{\mathsf{Iss}})$*;*
  - π_{del} *is a valid proof for the value* Δ_{ID} *in* del*;*
  - scope *included in* del*, which is part of* pres*, is satisfied.*

  If these checks pass, the delegated presentation is accepted, and the algorithm outputs 1.

Figure 2 describes the interaction flow between the delegator D, the delegatee Δ and the Verifier V.

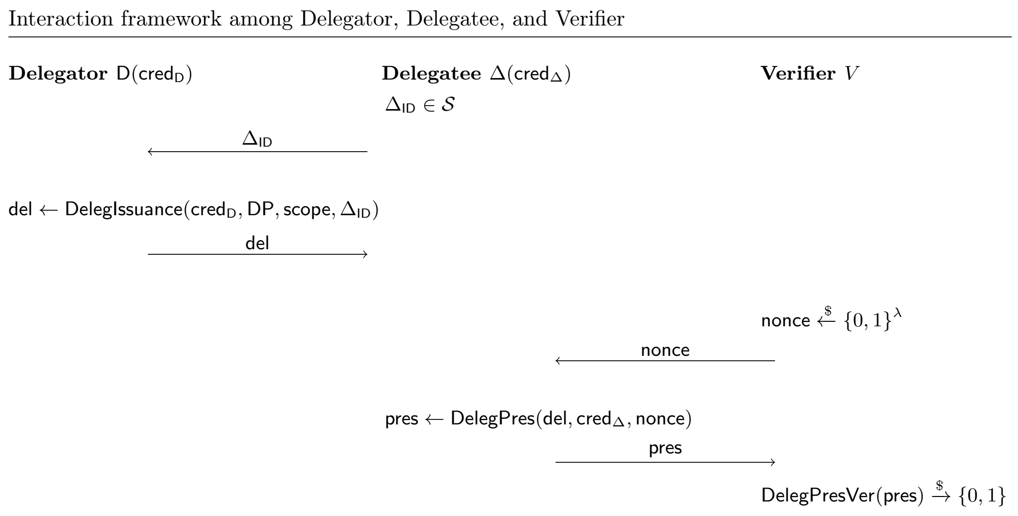

Interaction framework among Delegator, Delegatee, and Verifier
Delegator $\mathsf{D}(\mathsf{cred}_{\mathsf{D}})$
Delegatee $\Delta(\mathsf{cred}_{\Delta})$
Verifier V
$\Delta_{\mathsf{ID}} \in \mathcal{S}$
Δ_{ID}
$\mathsf{del} \leftarrow \mathsf{DelegIssuance}(\mathsf{cred}_{\mathsf{D}}, \mathsf{DP}, \mathsf{scope}, \Delta_{\mathsf{ID}})$
del
$\mathsf{nonce} \xleftarrow{\$} \{0,1\}^{\lambda}$
nonce
$\mathsf{pres} \leftarrow \mathsf{DelegPres}(\mathsf{del}, \mathsf{cred}_{\Delta}, \mathsf{nonce})$
pres
$\mathsf{DelegPresVer}(\mathsf{pres}) \xrightarrow{\$} \{0,1\}$

Fig. 2. Interactions between the delegator D, the delegatee Δ, and the Verifier V. According to the scenario, the delegatee might send to the delegator some information Δ_{ID} about its identity, so that the delegator can use it as an input for the algorithm DelegIssuance.

We now define the security properties for VP delegation schemes.

Correctness Property. The correctness property specifies that a delegation algorithm always allows an honest delegator (who generates a delegation for a DP consistent with its identity), to generate a delegation del and delegate another user. Then, if the user is an honest delegatee (its identity satisfies Δ_{ID}), it can always present del satisfying scope to a verifier.

Definition 11 (Correctness of VP delegation scheme). *Given a VP delegation scheme*

$$\mathcal{VPDS} = (\mathsf{DelegIssuance}, \mathsf{DelegVer}, \mathsf{DelegPres}, \mathsf{DelegPresVer}),$$

we say that the scheme is correct if $1 \leftarrow \mathsf{DelegPresVer}(\mathsf{pres})$ *whenever:*

– $\mathsf{del} \overset{\$}{\leftarrow} \mathsf{DelegIssuance}(\mathsf{cred}_\mathsf{D}, \mathsf{DP}, \mathsf{scope}, \varDelta_\mathsf{ID})$ *where* cred_D *satisfies the statements contained in* DP *(which is contained in* del*);*
– $\mathsf{pres} \overset{\$}{\leftarrow} \mathsf{DelegPres}(\mathsf{del}, \mathsf{cred}_\varDelta, \mathsf{nonce})$*, where* $\mathsf{cred}_\varDelta$ *satisfies the statements contained in* $\varDelta_\mathsf{ID}$*.*

Unforgeability Property. We consider two notions of unforgeability: the unforgeability of the delegation algorithm $\mathsf{DelegIssuance}$, which means that an adversary cannot forge a delegation from a credential it does not possess, and the unforgeability of the delegation presentation algorithm $\mathsf{DelegPres}$, which means that an adversary cannot present a delegation that is not issued to its identity (i.e., its identity does not satisfy $\varDelta_\mathsf{ID}$). We capture these two notions of unforgeability in a single experiment.

The experiment captures that an adversary $\mathcal{A}$, after receiving the public key of the issuer pk_Iss, the public parameters pp, and performing a training, cannot forge a delegation presentation. The training considers an adversary that can learn information from the system in the following ways: it can (1) corrupt a polynomial number of users, (2) receive a polynomial number of delegations from users it does not control, and (3) verify a polynomial number of delegated presentations (i.e., receive a polynomial number of presentations of its choice). These operations are modeled by allowing $\mathcal{A}$ to query a polynomial number of credentials to the oracle O_iss, of delegations to the oracle O_del and of presentations of delegations to the oracle O_pres.

Note that in a system supporting the delegation of VPs, the adversary can see not only delegated presentations, but also presentations of verifiable credentials (generated using the algorithm $\mathsf{CredPresentation}(\mathsf{cred}, \mathsf{stmt})$). However, we omit the queries for credentials presentations because we assume that credentials presentations can be seen as delegated presentations associated to an empty delegation $(\mathsf{del}, \mathsf{nonce}) = (\perp, \perp)$.

Experiment 1 ($\mathsf{Exp}_\mathcal{A}^{\mathsf{DelPresUnforgeability}}(1^\lambda)$) *The experiment consists of the following phases.*

Setup phase $\mathcal{A}$ *receives from the challenger of the experiment the public key* pk_Iss *of the issuer and the public parameters* pp *of the underlying VC or AC scheme.*

Training phase $\mathcal{A}$ *can interact with random oracles* $\mathsf{O}_\mathsf{iss}, \mathsf{O}_\mathsf{del}$ *and* O_pres*, to which it can send a polynomial number of issuing queries. Each query has the following input-output specification:*

1. *Issuing queries to* O_iss*:* $\mathcal{A}$ *can query for the issuance of a credential for a set of attributes* $\{a_i\}_{i\in[l]}$ *of its choice; the oracle computes* $\mathsf{cred} \leftarrow \mathsf{CredIssuance}(\{a_i\}_{i\in[l]}, \mathsf{sk}_\mathsf{Iss})$*, stores* cred *to the* credential table CT *and sends it to* $\mathcal{A}$*.*
2. *Delegation queries to* O_del*:* $\mathcal{A}$ *can query* O_del *for the issuance of a delegation for* $(\varDelta_\mathsf{ID}, \mathsf{scope}, \mathsf{DP})$ *of its choice; the oracle computes* $\mathsf{del} \leftarrow (\varDelta_\mathsf{ID}, \mathsf{scope}, \mathsf{DP}, \pi_\mathsf{DP})$*, stores* del *in the* delegation table DT *and sends it to* $\mathcal{A}$*.*
3. *Delegated presentation queries to* O_pres*:* $\mathcal{A}$ *can query for the presentation of a delegation for the values* $(\mathsf{del}, \mathsf{nonce})$ *of its choice; the oracle* O_pres *generates a presentation* pres *for* $(\mathsf{del}, \mathsf{nonce})$*, stores* pres *to the* presentation table PT *and sends it to* $\mathcal{A}$*.*

Forgery phase $\mathcal{A}$ *eventually outputs a delegated presentation* $\mathsf{pres}^\star = (\mathsf{del}^\star, \pi_{\mathsf{del}^\star})$*.*

Winning conditions *Parse* $\mathsf{pres}^\star = (\mathsf{del}^\star, \pi_{\mathsf{del}^\star})$ *as* $({\Delta_{\mathsf{ID}}}^\star, \mathsf{scope}^\star, \mathsf{DP}^\star, \pi_{\mathsf{DP}^\star}, \pi_{\mathsf{del}^\star})$.

The adversary wins the experiment if $1 \leftarrow \mathsf{DelegPresVer}(\mathsf{pres}^\star)$ *and at least one of the following conditions is satisfied:*

1. forgery of the delegation*:* $\mathsf{del}^\star = ({\Delta_{\mathsf{ID}}}^\star, \mathsf{scope}^\star, \mathsf{DP}^\star, \pi_{\mathsf{DP}^\star})$ *is forged, i.e.*
   - *it is not a delegation queried by* $\mathcal{A}$ *to* $\mathsf{O}_{\mathsf{del}}$*, i.e.* $\mathsf{del}^\star \notin \mathsf{DT}$[8]*;*
   - $\mathsf{del}^\star$ *is not generated using any credential issued to* $\mathcal{A}$[9].
2. forgery of the delegated presentation*:* $\pi_{\mathsf{del}^\star}$ *is forged, i.e.*
   - $\forall \mathsf{pres} \in \mathsf{PT}$*,* $\mathsf{pres} = (\mathsf{del}, \pi_{\mathsf{del}})$*,* $\pi_{\mathsf{del}} \neq \pi_{\mathsf{del}^\star}$*;*
   - $\pi_{\mathsf{del}^\star}$ *is not generated using any credential issued to* $\mathcal{A}$.

Definition 12 (Unforgeability of VP delegation scheme). *We say that a VP delegation scheme is unforgeable if for any PPT adversary* $\mathcal{A}$ *executing* $\mathsf{Exp}_{\mathcal{A}}^{\mathsf{DelPresUnforgeability}}(1^\lambda)$,

$$\Pr\left[\mathcal{A} \textit{ wins } \mathsf{Exp}_{\mathcal{A}}^{\mathsf{DelPresUnforgeability}}(1^\lambda)\right] \leq \nu(\lambda),$$

where $\nu(\lambda)$ *is negligible in the security parameter* λ.

Unlinkability Property. We consider two notions of *unlinkability*. The first concerns the $\mathsf{DelegIssuance}$ algorithm, ensuring that, given a delegation generated from one of two possible delegator credentials, an adversary cannot distinguish which credential was used. The second relates to the $\mathsf{DelegPres}$ algorithm, which guarantees that, given a delegated presentation generated from one of two possible delegatee credentials, an adversary cannot determine which credential produced it. We formalize these two notions of unlinkability through two corresponding experiments.

Experiment 2 ($\mathsf{Exp}_{\mathcal{A}}^{\mathsf{DelegIssuanceUnlinkability}}(1^\lambda)$)

1. *The adversary* $\mathcal{A}$ *generates the public parameters* pp *for the verifiable credential scheme together with the issuer key pair* $(\mathsf{sk}_{\mathsf{Iss}}, \mathsf{pk}_{\mathsf{Iss}})$ *and send* pp *and* $\mathsf{pk}_{\mathsf{Iss}}$ *to the challenger.*
2. *The challenger samples a random bit* $b \xleftarrow{\$} \{0,1\}$.
3. $\mathcal{A}$ *generates and sends to the challenger* Δ_{ID}*,* DP*,* scope *and* cred_{D_0} *and* cred_{D_1}*, satisfying* DP.
4. *The challenger runs* $\mathsf{del}_b \xleftarrow{\$} \mathsf{DelegIssuance}(\mathsf{cred}_{D_b}, \mathsf{DP}, \mathsf{scope}, \Delta_{\mathsf{ID}})$.
5. *The challenger sends* del_b *to* $\mathcal{A}$.
6. $\mathcal{A}$ *outputs a bit* b'.
7. $\mathcal{A}$ *wins if* $b' = b$.

If the adversary has access to a random oracle (according to the VP delegation scheme definition), it can send random oracle queries before sending to the challenger $\mathsf{del}, \mathsf{nonce}$*,* $\mathsf{cred}_{\mathsf{D}_0}$ *and* $\mathsf{cred}_{\mathsf{D}_1}$.

[8] In the constructions we will present, this implies that $\pi_{\mathsf{DP}^\star}$ is different from any π_{DP} generated by $\mathsf{O}_{\mathsf{del}}$.

[9] It might not be fully clear at this stage what this sentence means. The specific operations to perform this check change according to the kind of VC scheme we are considering. When we analyse the security of our VP delegation scheme, we provide more details.

Definition 13 (Unlinkability of the delegation). *A VP delegation scheme satisfies* unlinkability of the delegation *if for any PPT adversary $\mathcal{A}$ executing* $\mathsf{Exp}_{\mathcal{A}}^{\mathsf{DelegIssuanceUnlinkability}}(1^\lambda)$,

$$\left|\Pr\left[\mathcal{A} \textit{ wins } \mathsf{Exp}_{\mathcal{A}}^{\mathsf{DelegIssuanceUnlinkability}}(1^\lambda)\right] - \frac{1}{2}\right| = \left|\Pr[b = b'] - \frac{1}{2}\right| \leq \nu(\lambda),$$

where $\nu(\lambda)$ is negligible in the security parameter λ.

Experiment 3 ($\mathsf{Exp}_{\mathcal{A}}^{\mathsf{DelegPresUnlinkability}}(1^\lambda)$)

1. *The adversary $\mathcal{A}$ generates the public parameters* pp *for the verifiable credential scheme together with the issuer key pair* $(\mathsf{sk_{Iss}}, \mathsf{pk_{Iss}})$ *and send* pp *and* $\mathsf{pk_{Iss}}$ *to the challenger.*
2. *The challenger samples a random bit $b \xleftarrow{\$} \{0,1\}$.*
3. *The adversary $\mathcal{A}$ generates and sends to the challenger* del, nonce *and* cred_{Δ_0} *and* cred_{Δ_1}*, satisfying* Δ_{ID}.
4. *The challenger runs* $\mathsf{pres}_b \xleftarrow{\$} \mathsf{DelegPres}(\mathsf{del}, \mathsf{cred}_{\Delta_b}, \mathsf{nonce})$.
5. *The challenger sends* pres_b *to $\mathcal{A}$.*
6. *$\mathcal{A}$ outputs a bit $\bar{b}$.*
7. *$\mathcal{A}$ wins if $\bar{b} = b$.*

If the adversary has access to a random oracle (according to the VP delegation scheme definition), it can send random oracle queries before sending to the challenger $\mathsf{del}, \mathsf{nonce}$, cred_{Δ_0} *and* cred_{Δ_1}.

Definition 14 (Unlinkability of the delegated presentation). *A VP delegation scheme satisfies* unlinkability of the delegated presentation *if for any PPT adversary $\mathcal{A}$ executing* $\mathsf{Exp}_{\mathcal{A}}^{\mathsf{DelegPresUnlinkability}}(1^\lambda)$,

$$\left|\Pr\left[\mathcal{A} \textit{ wins } \mathsf{Exp}_{\mathcal{A}}^{\mathsf{DelegPresUnlinkability}}(1^\lambda)\right] - \frac{1}{2}\right| = \left|\Pr[\bar{b} = b] - \frac{1}{2}\right| \leq \nu(\lambda),$$

where $\nu(\lambda)$ is negligible in the security parameter λ.

5 VP Delegation Schemes from mdoc VCs and BBS ACs

In this section, we first create an instance of a VP delegation scheme according to Definition 10 that is built on top of the mdoc VC scheme, as described in Definition 7. Then, we apply the same design choices to create an instance of a VP delegation scheme built on top of the BBS AC scheme, as described in Definition 9.

VP Delegation Scheme from mdoc VCs. In the former case, the delegator and the delegatee each have a VC issued by the issuer of the following form:

$$\mathsf{cred_D} = \left((\sigma_\mathsf{D}, \{\mathsf{com}_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred_D}}), \{a_{\mathsf{D},i}\}_{i\in[l]}, \{\mathsf{salt}_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{sk}_{\mathsf{cred_D}}\right),$$

$$\mathsf{cred}_\Delta = \left((\sigma_\Delta, \{\mathsf{com}_{\Delta,i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\Delta}), \{a_{\Delta,i}\}_{i\in[l]}, \{\mathsf{salt}_{\Delta,i}\}_{i\in[l]}, \mathsf{sk}_{\mathsf{cred}_\Delta}\right).$$

Note that in this case all the proofs introduced in Definition 10 are digital signatures.

In the latter case, instead, the delegator and the delegatee each have a BBS credential issued by the issuer of the following form:

$$\mathsf{cred}_\mathsf{D} = \left(\sigma_\mathsf{D}, \{a_{\mathsf{D},i}\}_{i\in[l]}\right),$$

$$\mathsf{cred}_\Delta = \left(\sigma_\Delta, \{a_{\Delta,i}\}_{i\in[l]}\right).$$

where $\sigma_\mathsf{D} = (A_\mathsf{D}, e_\mathsf{D})$ and $\sigma_\Delta = (A_\Delta, e_\Delta)$. In this case, all the proofs are non-interactive zero-knowledge proofs.

Definition 15 (mdoc VP Delegation Scheme). *Given the mdoc VC scheme defined in Definition 7, which uses the digital signature scheme $\mathcal{DS}$ and a cryptographic hash function H, we can define the following VP delegation scheme (see Definition 10).*

– Delegation issuance*:* $\underbrace{(\Delta_\mathsf{ID}, \mathsf{scope}, \mathsf{DP}, \pi_\mathsf{DP})}_{\mathsf{del}} \overset{\$}{\leftarrow} \mathsf{DelegIssuance}(\mathsf{cred}_\mathsf{D}, \mathsf{DP}, \mathsf{scope}, \Delta_\mathsf{ID})$.
  On input $(\mathsf{cred}_\mathsf{D}, \mathsf{DP}, \mathsf{scope}, \Delta_\mathsf{ID})$*, the delegator computes* π_DP *as a variant of a presentation of* cred_D *which is bound to* DP *but also to* scope *and* Δ_ID*:*
  - $\mathsf{pres}' \leftarrow ((\sigma_\mathsf{D}, \{\mathsf{com}_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\mathsf{D}}), \{\mathsf{salt}_{\mathsf{D},i}\}_{i\in\mathsf{DP}})$*;*
  - $\sigma' \overset{\$}{\leftarrow} \mathsf{Sign}((\mathsf{pres}'||\mathsf{DP}, \mathsf{scope}, \Delta_\mathsf{ID}), \mathsf{sk}_{\mathsf{cred}_\mathsf{D}})$ *and* $\pi_\mathsf{DP} \leftarrow (\mathsf{pres}', \sigma')$*;*
  - *return* $\mathsf{del} \leftarrow ((\Delta_\mathsf{ID}, \mathsf{scope}, \mathsf{DP}, \pi_\mathsf{DP}))$.[10]

– Delegation verification*:* $\{0,1\} \overset{\$}{\leftarrow} \mathsf{DelegVer}(\mathsf{del}, \mathsf{pk}_\mathsf{Iss})$.
  To verify the delegation, parse:

$$\mathsf{del} \to (\Delta_\mathsf{ID}, \mathsf{scope}, \mathsf{DP}, \pi_\mathsf{DP}), \quad \pi_\mathsf{DP} \to (\mathsf{pres}', \sigma'),$$

$$\mathsf{pres}' \to ((\sigma_\mathsf{D}, \{\mathsf{com}_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\mathsf{D}}), \{\mathsf{salt}_{\mathsf{D},i}\}_{i\in\mathsf{DP}}).$$

  Then, perform the following checks:
  - *Verify the signature of the issuer:* $1 \leftarrow \mathsf{Vf}(\sigma_\mathsf{D}, (\{\mathsf{com}_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\mathsf{D}}), \mathsf{pk}_\mathsf{Iss})$*. This means that the delegation is created from a valid VC issued by* pk_Iss*;*
  - *Check that* $\mathsf{com}_{\mathsf{D},i} = H(a_{\mathsf{D},i}||\mathsf{salt}_{\mathsf{D},i}), \forall i \in \mathsf{DP}$*. This means that the delegation has a* DP *consistent with the credential used to generate it;*
  - *verify the signature* σ' *of* pres' *using the public key* $\mathsf{pk}_{\mathsf{cred}_\mathsf{D}}$*:*

$$1 \leftarrow \mathsf{Vf}(\sigma', (\mathsf{pres}'||\mathsf{DP}, \mathsf{scope}, \Delta_\mathsf{ID}), \mathsf{pk}_{\mathsf{cred}_\mathsf{D}}).$$

    This means that the delegation (for scope *and* Δ_ID*) was created by the holder of the associated credential.*

– Delegated presentation*:* $\underbrace{(\mathsf{del}, \pi_\mathsf{del})}_{\mathsf{pres}} \overset{\$}{\leftarrow} \mathsf{DelegPres}(\mathsf{del}, \mathsf{cred}_\Delta, \mathsf{nonce})$.
  After parsing $\mathsf{del} \to (\Delta_\mathsf{ID}, \mathsf{scope}, \mathsf{DP}, \pi_\mathsf{DP})$*, the delegatee computes* π_del *as a presentation of* Δ_ID *using* cred_Δ *which is bound to* del*:*
  - *compute* $\mathsf{pres}'' \leftarrow ((\sigma_\Delta, \{\mathsf{com}_{\Delta,i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\Delta}), \{\mathsf{salt}_{\Delta,i}\}_{i\in\Delta_\mathsf{ID}}, \mathsf{nonce})$*;*
  - *the delegatee signs* pres'' *computing* $\sigma'' \overset{\$}{\leftarrow} \mathsf{Sign}((\mathsf{pres}''||\mathsf{del}), \mathsf{sk}_{\mathsf{cred}_\Delta})$*, sets* $\pi_\mathsf{del} \leftarrow (\mathsf{pres}'', \sigma'')$ *and returns* $\mathsf{pres} \leftarrow (\mathsf{del}, \pi_\mathsf{del})$.

[10] Note that π_DP is structured as a presentation of the mdoc VC which is bound, via σ', to the specific $\mathsf{DP}, \mathsf{scope}, \Delta_\mathsf{ID}$ that will be part of the delegation del.

– Delegated presentation verification*:* $\{0,1\} \xleftarrow{\$} \mathsf{DelegPresVer}(\mathsf{pres}, \mathsf{pk}_{\mathsf{Iss}})$.
Parse $\mathsf{pres} \rightarrow (\mathsf{del}, \pi_{\mathsf{del}})$*, where* $\pi_{\mathsf{del}} \rightarrow (\mathsf{pres}'', \sigma'')$ *and*
$\mathsf{pres}'' \rightarrow ((\sigma_\Delta, \{\mathsf{com}_{\Delta,i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\Delta}), \{\mathsf{salt}_{\Delta,i}\}_{i\in\Delta_{\mathsf{ID}}}), \mathsf{nonce})$. *The verifier checks that*
  - *the delegation is valid, i.e.* $1 \leftarrow \mathsf{DelegVer}(\mathsf{del})$*;*
  - π_{del} *is a valid presentation of* cred_Δ *for the attributes in* Δ_{ID} *specified in* del*, i.e.:*
    1. *the signature of the issuer is valid:* $1 \leftarrow \mathsf{Vf}(\sigma_\Delta, (\{\mathsf{com}_{\Delta,i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\Delta}), \mathsf{pk}_{\mathsf{Iss}})$.
    2. $\mathsf{com}_{\Delta,i} = H(a_{\Delta,i}||\mathsf{salt}_{\Delta,i})\,\forall i \in \Delta_{\mathsf{ID}}$*;*
    3. *the signature* σ'' *of* pres'' *is valid using* $\mathsf{pk}_{\mathsf{cred}_\Delta}$*:* $1 \leftarrow \mathsf{Vf}(\sigma'', (\mathsf{pres}''||\mathsf{del}), \mathsf{pk}_{\mathsf{cred}_\Delta})$*;*
  - *the value* scope *included in* del *is satisfied according to the associated verification procedure.*

If these checks pass, the delegated presentation is accepted, and the algorithm outputs 1.

VP Delegation Scheme from BBS ACs. To clearly define the instantiation of our BBS-based delegation scheme, and to explicitly describe the attributes revealed and kept hidden by the delegator and the delegatee, we introduce the following notation:

– $\mathsf{Rev}_{\mathsf{D}}$ as the set of indices of the attributes disclosed by the delegator, identified by DP.
– $\mathsf{Hid}_{\mathsf{D}} = [\ell]\setminus\mathsf{Rev}_{\mathsf{D}}$ as the set of indices corresponding to the delegator's hidden attributes.
– Rev_Δ as the set of indices of the attributes disclosed by the delegatee, identified by Δ_{ID}.
– $\mathsf{Hid}_\Delta = [\ell]\setminus\mathsf{Rev}_\Delta$ as the set of indices corresponding to the delegatee's hidden attributes.

Definition 16 (BBS VP Delegation Scheme).
Given the BBS AC scheme defined in Definition 9, we can define the following delegation scheme (see Definition 10).

– Delegation issuance*:* $\underbrace{(\Delta_{\mathsf{ID}}, \mathsf{scope}, \mathsf{DP}, \pi_{\mathsf{DP}})}_{\mathsf{del}} \xleftarrow{\$} \mathsf{DelegIssuance}(\mathsf{cred}_{\mathsf{D}}, \mathsf{DP}, \mathsf{scope}, \Delta_{\mathsf{ID}})$.
On input $(\mathsf{cred}_{\mathsf{D}}, \mathsf{DP}, \mathsf{scope}, \Delta_{\mathsf{ID}})$ *the delegator* D *computes a non-interactive zero-knowledge proof* π_{DP}*, which is bound to* DP*,* scope *and* Δ_{ID}*, to prove that* DP *is included in* $\mathsf{anoncred}_D$*:*
  1. *generates the commitment of the underlying sigma protocol* $(\bar{A}_{\mathsf{D}}, \bar{B}_{\mathsf{D}}, U_{\mathsf{D}})$*:*
     - $r_1 \leftarrow \mathbb{Z}_p^*$*;*
     - $\bar{A}_{\mathsf{D}} \leftarrow A_{\mathsf{D}}^{r_1}, \quad \bar{B}_{\mathsf{D}} \leftarrow C(\mathbf{a}_{\mathsf{D}})^{r_1}\bar{A}_{\mathsf{D}}^{-e_{\mathsf{D}}}$*;*
     - $\alpha_1, \beta_1 \leftarrow \mathbb{Z}_p, \quad \delta_{1,i} \leftarrow \mathbb{Z}_p \quad \forall\, i \in \mathsf{Hid}_{\mathsf{D}}$*;*
     - $U_{\mathsf{D}} \leftarrow \bar{A}_{\mathsf{D}}^{\alpha_1}\bar{B}_{\mathsf{D}}^{\beta_1}\prod_{i\in\mathsf{Hid}_{\mathsf{D}}}\mathbf{h}[i]^{\delta_{1,i}}$.
  2. *computes the challenge* $c_1 \leftarrow H(\bar{A}_{\mathsf{D}}, \bar{B}_{\mathsf{D}}, U_{\mathsf{D}}, \mathsf{DP}, \mathsf{scope}, \Delta_{\mathsf{ID}})$*;*
  3. *computes a response* z_1*:*
     - $s_1 \leftarrow \alpha_1 + r_1^{-1}\cdot e_{\mathsf{D}}\cdot c_1$*;*
     - $n_1 \leftarrow \beta_1 + r_1^{-1}\cdot c_1$*;*
     - $u_{1,i} \leftarrow \delta_{1,i} - \mathbf{a}_{\mathsf{D}}[i]\,c_1 \quad \forall\, i \in \mathsf{Hid}_{\mathsf{D}}$*;*
     - $z_1 \leftarrow (s_1, n_1, (u_{1,i})_{i\in\mathsf{Hid}_{\mathsf{D}}})$.
  4. *sets* $\pi_{\mathsf{DP}} \leftarrow (\bar{A}_{\mathsf{D}}, \bar{B}_{\mathsf{D}}, c_1, z_1)$*;*[11]
  5. *returns* $\mathsf{del} \leftarrow (\Delta_{\mathsf{ID}}, \mathsf{scope}, \mathsf{DP}, \pi_{\mathsf{DP}})$.

– Delegation verification*:* $\{0,1\} \xleftarrow{\$} \mathsf{DelegVer}(\mathsf{del}, \mathsf{pk}_{\mathsf{Iss}})$.
To verify the delegation, parse:

$$\mathsf{del} \rightarrow (\Delta_{\mathsf{ID}}, \mathsf{scope}, \mathsf{DP}, \pi_{\mathsf{DP}}), \quad \pi_{\mathsf{DP}} \rightarrow (\bar{A}_{\mathsf{D}}, \bar{B}_{\mathsf{D}}, c_1, z_1).$$

Then, the delegatee:

[11] Note that from $\bar{A}_{\mathsf{D}}, \bar{B}_{\mathsf{D}}, c_1$ and z_1 the verifier can reconstruct U_D.

1. *computes* $\tilde{U}_{\mathsf{D}}$
$$\tilde{U}_{\mathsf{D}} \leftarrow \bar{A}_{\mathsf{D}}^{s_1} \bar{B}_{\mathsf{D}}^{n_1} \prod_{i \in \mathsf{Hid}_{\mathsf{D}}} \mathbf{h}[i]^{u_{1,i}} C_{\mathsf{Rev}_{\mathsf{D}}}(\mathbf{a}_{\mathsf{D}})^{-c_1};$$
2. *checks if*
$$c_1 \overset{?}{=} H(\bar{A}_{\mathsf{D}}, \bar{B}_{\mathsf{D}}, \tilde{U}_D, \mathsf{DP}, \mathsf{scope}, \Delta_{\mathsf{ID}}),$$
$$\mathbf{e}(\bar{A}_{\mathsf{D}}, X_2) \overset{?}{=} \mathbf{e}(\bar{B}_{\mathsf{D}}, g_2).$$

– Delegated presentation*:* $\underbrace{(\mathsf{del}, \pi_{\mathsf{del}})}_{\mathsf{pres}} \overset{\$}{\leftarrow} \mathsf{DelegPres}(\mathsf{del}, \mathsf{cred}_{\Delta}, \mathsf{nonce})$*.*
After receiving the nonce*, the delegatee computes a non-interactive zero-knowledge proof* π_{del}*, which is bound to* del*, to prove that* Δ_{ID} *is included in* cred_{Δ}*, in the following way:*
1. *generates the commitment of the underlying sigma protocol* $(\bar{A}_{\Delta}, \bar{B}_{\Delta}, U_{\Delta})$*:*
   - $r_2 \leftarrow \mathbb{Z}_p^*$*;*
   - $\bar{A}_{\Delta} \leftarrow A_{\Delta}^{r_2}, \quad \bar{B}_{\Delta} \leftarrow C(\mathbf{a}_{\Delta})^{r_2} \bar{A}_{\Delta}^{-e_{\Delta}}$*;*
   - $\alpha_2, \beta_2 \leftarrow \mathbb{Z}_p, \quad \delta_{2,i} \leftarrow \mathbb{Z}_p \quad \forall\, i \in \mathsf{Hid}_{\Delta}$*;*
   - $U_{\Delta} \leftarrow \bar{A}_{\Delta}^{\alpha_2} \bar{B}_{\Delta}^{\beta_2} \prod_{i \in \mathsf{Hid}_{\Delta}} \mathbf{h}[i]^{\delta_{2,i}}$*.*
2. *computes the challenge* $c_2 \leftarrow H(\bar{A}_{\Delta}, \bar{B}_{\Delta}, U_{\Delta}, \mathsf{del}, \mathsf{nonce})$*;*
3. *computes a response* z_2*:*
   - $s_2 \leftarrow \alpha_2 + r_2^{-1} \cdot e_{\Delta} \cdot c_2$*;*
   - $n_2 \leftarrow \beta_2 + r_2^{-1} \cdot c_2$*;*
   - $u_{2,i} \leftarrow \delta_{2,i} - \mathbf{a}_{\Delta}[i]\, c_2 \quad \forall\, i \in \mathsf{Hid}_{\Delta}$*;*
   - $z_2 \leftarrow (s_2, n_2, (u_{2,i})_{i \in \mathsf{Hid}_{\Delta}})$*.*
4. *sets* $\pi_{\mathsf{del}} \leftarrow (\bar{A}_{\Delta}, \bar{B}_{\Delta}, c_2, z_2)$
5. *returns* $\mathsf{pres} \leftarrow (\mathsf{del}, \pi_{\mathsf{del}})$*.*

– Delegated presentation verification*:* $\{0,1\} \overset{\$}{\leftarrow} \mathsf{DelegPresVer}(\mathsf{pres}, \mathsf{pk}_{\mathsf{Iss}})$*.*
Parse
$$\mathsf{pres} \to (\mathsf{del}, \pi_{\mathsf{del}}), \quad \mathsf{del} \to (\Delta_{\mathsf{ID}}, \mathsf{scope}, \mathsf{DP}, \pi_{\mathsf{DP}}),$$
$$\pi_{\mathsf{DP}} \to (\bar{A}_{\mathsf{D}}, \bar{B}_{\mathsf{D}}, c_1, z_1), \qquad \pi_{\mathsf{del}} = (\bar{A}_{\Delta}, \bar{B}_{\Delta}, c_2, z_2).$$
The verifier checks that
1. *the delegation is valid, i.e.* $1 \leftarrow \mathsf{DelegVer}(\mathsf{del}, \mathsf{pk}_{\mathsf{Iss}})$*;*
2. π_{del} *is a valid proof for the attributes in* Δ_{ID} *specified in* del*:*
   - *computes* $\tilde{U}_{\Delta}$*:*
   $$\tilde{U}_{\Delta} \leftarrow \bar{A}_{\Delta}^{s_2} \bar{B}_{\Delta}^{n_2} \prod_{i \in \mathsf{Hid}_{\Delta}} \mathbf{h}[i]^{u_{2,i}} C_{\mathsf{Rev}_{\Delta}}(\mathbf{a}_{\Delta})^{-c_2};$$
   - *checks that*
   $$c_2 \overset{?}{=} H(\bar{A}_{\Delta}, \bar{B}_{\Delta}, \tilde{U}_{\Delta}, \mathsf{del}, \mathsf{nonce}),$$
   $$\mathbf{e}(\bar{A}_{\Delta}, X_2) \overset{?}{=} \mathbf{e}(\bar{B}_{\Delta}, g_2).$$
3. *the value* scope *included in* π_{DP} *is satisfied according to the associated verification procedure.*

If these checks pass, the delegated presentation is accepted, and the algorithm outputs 1.

6 Security Analysis

In this section, we show that both the scheme built on top of mdoc VCs (Definition 15) and BBS ACs (Definition 16) satisfy the correctness and unforgeability properties[12] (Definition 11 and Definition 12, respectively). Then we show that the VP delegation scheme built on top of the BBS ACs also satisfies the unlinkability of the delegated presentation property (Definition 14). The unlinkability of the delegation issuance (Definition 13) can be proved using a similar argument.

6.1 Correctness

Theorem 1. *The delegation scheme described in Definition 15 instantiated using the digital signature $\mathcal{DS}$ is correct, assuming that the digital signature $\mathcal{DS}$ is correct and that H is modeled as a random oracle.*

Proof. It is easy to see that the VP delegation scheme is correct. In fact, the delegator always computes a proof π_{DP} to include in del, and, since the digital signature $\mathcal{DS}$ is correct, del is a valid delegation. Then, if the delegation is presented by a delegatee whose identity matches with the identity Δ_{ID} included in del, the delegatee can always successfully present the delegation, again because $\mathcal{DS}$ is correct.

Theorem 2. *The delegation scheme described in Definition 16 instantiated using the digital signature BBS is correct, assuming that H is modeled as a random oracle.*

Proof. In this case, the correctness follows directly from the correctness property of the underlying NIZKP for BBS signatures.

6.2 Unforgeability

We observe that in the definition of the Experiment 1, it is not well defined what we mean by $\mathsf{del}^\star$ (or $\pi_{\mathsf{del}^\star}$) "is not generated using any credential issued to $\mathcal{A}$". In fact, this sentence has a different meaning according to the amount of information a presentation reveals about the credential used to generate it. More specifically, if we consider the delegation of presentation designed on top of mdoc VCs (as defined in Definition 15), the presentation and delegation contain an identifier of the VC used to generate them, namely $\mathsf{pk}_{\mathsf{cred}}$, and all the commitments to the attributes. Therefore, in this case, we say that the presentation is not generated using any credential issued to $\mathcal{A}$ if the information that identifies the credential used to generate the presentation does not match any of the credentials issued to $\mathcal{A}$. For the delegation of anonymous presentations, as those built on top of BBS credential (as defined in Definition 16), the meaning is different: since the presentation hides the credential used to generate it, and the only link between the presentation (or delegation) and the credential used to generate it are the issuer of the credential $\mathsf{pk}_{\mathsf{Iss}}$ and the attribute revealed, we say that a forgery has not been generated using any credential issued to $\mathcal{A}$ if the adversary has never been issued a credential by $\mathsf{pk}_{\mathsf{Iss}}$ for the attributes revealed in the forgery.

In the security proof of unforgeability of the delegation scheme for mdoc VCs (as defined in Definition 15), we must introduce a technicality that simplifies the description of our

[12] In this regard, it is worth noting how the analysis to prove the unforgeability proof for the delegation scheme for BBS ACs is way simpler than the analysis presented for mdoc VCs.

reduction. In particular, we show that the adversary $\mathcal{A}$ to the unforgeability of the VP delegation scheme can be used to build a reduction to a *variant* of the strong unforgeability under chosen message attacks (SUF-CMA) of the digital signature $\mathcal{DS}$. In this variant, which we refer to as *v-SUF-CMA*, the reduction can open a polynomial number of sessions of standard SUF-CMA experiments (each associated with a different public key with the same public parameters pp), and the reduction wins the v-SUF-CMA experiment if it produces a forgery for at least one of the public keys provided by the challenger of the experiment. Informally, the queries that the adversary can make are (1) queries for the opening of a new session identified by a counter i, for which it receives a public key pk_i, and (2) signing queries specifying a message and one of the public keys it has previously received (m, pk_j), for which it receives a valid signature σ of m under the public key pk_j. It is easy to see that an adversary who can win the SUF-CMA experiment can be turned into an adversary of the v-SUF-CMA experiment, and the success probability remains the same. However, it is also easy to see that an adversary of v-SUF-CMA can be used as a subroutine for a reduction to the SUF-CMA with a loss in the probability of success proportional to the number of sessions the adversary can open.

This technicality is needed to capture that an adversary of the delegation scheme for mdoc VCs could win the experiment by forging the public key of the issuer, or by forging the credential public keys, for which it does not know the secret counterpart, that it sees during the unforgeability experiment training (for example the public keys included in the delegations queried to $\mathsf{O}_{\mathsf{del}}$ and in the presentations queried to $\mathsf{O}_{\mathsf{pres}}$). Therefore, this issue does not arise when evaluating the security of the delegation scheme for BBS anonymous credentials (where π_{del} and π_{pres} are NIZKP of a BBS credential under $\mathsf{pk}_{\mathsf{Iss}}$ and do not contain any other pk), because it reduces to the strong unforgeability of the BBS signature scheme instantiated with the public key $\mathsf{pk}_{\mathsf{Iss}}$. This remarkably simplifies the security analysis of VP delegation schemes built on top of BBS ACs, compared to the mdoc VCs.

Theorem 3. *The delegation scheme described in Definition 15 is unforgeable according to Definition 12 assuming that the digital signature $\mathcal{DS}$ is strongly v-UF-CMA (v-SUF-CMA) unforgeable and that H is a collision resistant cryptographic hash function*[13].

Proof Sketch. We prove that if there exists an adversary $\mathcal{A}$ of Experiment 1 who can win the experiment with non-negligible probability, then we can use $\mathcal{A}$ to build a reduction $\mathcal{B}$ to the v-SUF-CMA experiment for the underlying digital signature scheme $\mathcal{DS}$, which wins the experiment with non-negligible probability, or that breaks the collision resistance of the underlying hash function.

Setup $\mathcal{B}$ sends a query for a new public key in the v-SUF-CMA experiment. $\mathcal{C}$ sends to $\mathcal{B}$ a public key pk and public parameters pp. $\mathcal{B}$ sets $\mathsf{pk}_{\mathsf{Iss}} \leftarrow \mathsf{pk}$, and forwards $(\mathsf{pp}, \mathsf{pk}_{\mathsf{Iss}})$ to the adversary $\mathcal{A}$.

Training phase We show how $\mathcal{B}$ answers to the queries $\mathcal{A}$ can send during the training phase.

– Queries to $\mathsf{O}_{\mathsf{iss}}$: $\mathcal{A}$ sends in input a set of attributes $\{a_i\}_{i\in[l]}$. $\mathcal{B}$ performs the following operations:

[13] To prove unforgeability we need the commitment scheme used in mdoc VCs (Definition 7) to be binding. The hiding property is useful to enable the selective disclosure of attributes.

1. it generates a key pair $(\mathsf{sk}_{\mathsf{cred}}, \mathsf{pk}_{\mathsf{cred}})$[14], the salts $\{\mathsf{salt}_i\}_{i\in[l]}$, computes a commitment $\mathsf{com}_i = \mathsf{RO}(a_i||\mathsf{salt}_i)$ (where RO is a random oracle programmed by $\mathcal{B}$) to $a_i, \forall i \in [l]$;
2. it sends a signing query to $\mathcal{C}$ with input $((\{\mathsf{com}_i\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}}), \mathsf{pk}_{\mathsf{Iss}})$[15] : $\mathcal{C}$ returns a signature σ of $(\{\mathsf{com}_i\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}})$ using $\mathsf{sk}_{\mathsf{Iss}}$.
3. sends to $\mathcal{A}$ the credential

$$\mathsf{cred} \leftarrow ((\sigma, \{\mathsf{com}_i\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}}), \{a_i\}_{i\in[l]}, \{\mathsf{salt}_i\}_{i\in[l]}, \mathsf{sk}_{\mathsf{cred}})$$

and adds cred to CT.

– Queries to $\mathsf{O}_{\mathsf{del}}$: $\mathcal{A}$ can query for a delegation for the values $(\Delta_{\mathsf{ID}}, \mathsf{scope}, \mathsf{DP})$ of its choice. Upon receiving a query $\mathcal{B}$ performs the following operations:
  1. generates a set of random attributes $\{a_{\mathsf{D},i}\}_{i\in[l]}$ which satisfies DP, and computes the commitments com_i as before generating the salts and programming the random oracle;
  2. opens a new session of SUF-CMA with $\mathcal{C}$ from which it receives a public key $\mathsf{pk}_{\mathsf{cred}_\mathsf{D}}$ and sends a signing query $((\{\mathsf{com}_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\mathsf{D}}), \mathsf{pk}_{\mathsf{Iss}})$ for a signature with $\mathsf{sk}_{\mathsf{Iss}}$ of $(\{\mathsf{com}_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\mathsf{D}})$, and receives from $\mathcal{C}$ the signature σ_D;
  3. sends a signing query $((\mathsf{pres}', \mathsf{DP}, \mathsf{scope}, \Delta_{\mathsf{ID}}), \mathsf{pk}_{\mathsf{cred}_\mathsf{D}})$ to $\mathcal{C}$ for a signature with $\mathsf{sk}_{\mathsf{cred}_\mathsf{D}}$ of $\mathsf{pres}' = ((\sigma_\mathsf{D}, \{\mathsf{com}_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\mathsf{D}}), \{\mathsf{salt}_{\mathsf{D},i}\}_{i\in\mathsf{DP}})$. $\mathcal{C}$ returns the signature σ' ;
  4. $\mathcal{B}$ sets $\pi_{\mathsf{DP}} = (\sigma', \mathsf{pres}')$ sends to $\mathcal{A}$ the delegation $\mathsf{del} = (\Delta_{\mathsf{ID}}, \mathsf{scope}, \mathsf{DP}, \pi_{\mathsf{DP}})$ and adds del to DT.

Observation 1 *It is important that the reduction $\mathcal{B}$ does not choose the secret key $\mathsf{sk}_{\mathsf{cred}_\mathsf{D}}$ of the credential used to create the delegation* del *because the adversary $\mathcal{A}$ might create a forgery using the same public key* $\mathsf{pk}_{\mathsf{cred}}$ *and the reduction must be able to exploit that forgery.*

– Queries to $\mathsf{O}_{\mathsf{pres}}$: $\mathcal{A}$ can query for a delegated presentation giving in input $(\mathsf{del}, \mathsf{nonce})$ of its choice. Upon receiving a query $\mathcal{B}$ performs the following operations:
  1. generates a set of random attributes $\{a_{\Delta,i}\}_{i\in[l]}$ which satisfies Δ_{ID}, and computes the commitments com_i as before generating the salts and programming the random oracle;
  2. opens a new session of SUF-CMA with $\mathcal{C}$ from which it receives a public key $\mathsf{pk}_{\mathsf{cred}_\Delta}$ and sends a signing query $((\{\mathsf{com}_{\Delta,i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\Delta}), \mathsf{pk}_{\mathsf{Iss}})$ for a signature with $\mathsf{sk}_{\mathsf{Iss}}$ of $(\{\mathsf{com}_{\Delta,i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\Delta})$, and receives from $\mathcal{C}$ the signature σ;
  3. $\mathcal{B}$ sends a signing query $(\mathsf{pres}''||\mathsf{del}, \mathsf{pk}_{\mathsf{cred}_\Delta})$ to $\mathcal{C}$ or a signature with $\mathsf{sk}_{\mathsf{cred}_\Delta}$ of $\mathsf{pres}'' = ((\sigma_\Delta, \{\mathsf{com}_{\Delta,i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\Delta}), \{\mathsf{salt}_{\Delta,i}\}_{i\in\Delta_{\mathsf{ID}}}, \mathsf{nonce})$. $\mathcal{C}$ returns the signature σ''.
  4. $\mathcal{B}$ sends $\mathsf{pres} = (\mathsf{del}, (\sigma'', \mathsf{pres}''))$ to $\mathcal{A}$ and stores pres in PT.

Note that $\mathcal{B}$ knows only the secret keys associated to the credential issued to $\mathcal{A}$. The other $\mathsf{sk}_{\mathsf{cred}}$ are not known neither to $\mathcal{B}$, nor to $\mathcal{A}$. Therefore if $\mathcal{A}$ manages to forge the corresponding public keys $\mathsf{pk}_{\mathsf{cred}}$ that are included in the delegations or delegated presentations, the reduction can re-use such forgeries to win the v-SUF-CMA experiment.

[14] For clarity of presentation, the subscripts D (delegator) and Δ (delegatee) are omitted in the notation. The credential issuance procedure is the same for both roles and can be applied to either party.

[15] Note that this is the format for a signing query in the v-SUF-CMA experiment, given by the message to be signed, and the public key that will verify the signature.

Forgery Phase $\mathcal{A}$ sends to $\mathcal{B}$ a delegated presentation $\mathsf{pres}^\star = (\mathsf{del}^\star, \pi_{\mathsf{del}^\star})$ as its forgery. We argue that this presentation is a forgery, i.e. it satisfies at least one of the winning conditions of Experiment 1.

Instantiation and analysis of the winning condition of Experiment 1 We now focus on the winning condition of Experiment 1, and we rewrite it considering the experiment instantiated with the ARF-Compliant VP delegation Scheme in Definition 15.
The adversary's output is $\mathsf{pres}^\star = (\mathsf{del}^\star, \pi_{\mathsf{del}^\star})$, with $\mathsf{del}^\star = (\Delta_{\mathsf{ID}}{}^\star, \mathsf{scope}^\star, \mathsf{DP}^\star, \pi_{\mathsf{DP}^\star})$. We recall the structure of the proofs $\pi_{\mathsf{DP}^\star}$ and $\pi_{\mathsf{del}^\star}$.

1. $\pi_{\mathsf{DP}^\star}$ is a presentation of a credential for the attributes specified in $\mathsf{DP}^\star$ containing also $\mathsf{scope}^\star$ and $\Delta_{\mathsf{ID}}{}^\star$:
$$\pi_{\mathsf{DP}^\star} = \Bigg(\underbrace{((\sigma_{\mathsf{D}}, \{\mathsf{com}_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_{\mathsf{D}}}), \{\mathsf{salt}_{\mathsf{D},i}\}_{i\in\mathsf{DP}^\star})}_{\mathsf{pres}'},\ \sigma' = \mathsf{Sign}((\mathsf{pres}', \mathsf{DP}^\star, \mathsf{scope}^\star, \Delta_{\mathsf{ID}}{}^\star), \mathsf{sk}_{\mathsf{cred}_{\mathsf{D}}})\Bigg).$$
with σ' being the signature of pres' using the secret key associated to $\mathsf{pk}_{\mathsf{cred}_{\mathsf{D}}}$ and σ_{D} is the signature of $(\{\mathsf{com}_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_{\mathsf{D}}})$ (part of $\mathsf{cred}_{\mathsf{D}}$) using $\mathsf{sk}_{\mathsf{Iss}}$.
2. $\pi^\star_{\mathsf{del}}$ is a presentation of a credential cred_Δ for the attributes in Δ_{ID} bounded to $\mathsf{del}^\star$ and $\mathsf{nonce}^\star$, in particular:
$$\pi_{\mathsf{del}^\star} = \Bigg(\underbrace{((\sigma_\Delta, \{\mathsf{com}_{\Delta,i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\Delta}), \{\mathsf{salt}_{\Delta,i}\}_{i\in\Delta_{\mathsf{ID}}{}^\star}, \mathsf{nonce}^\star)}_{\mathsf{pres}''},\ \sigma'' = \mathsf{Sign}((\mathsf{pres}'', \mathsf{del}^\star), \mathsf{sk}_{\mathsf{cred}_\Delta})\Bigg).$$
with σ'' being the signature of pres'' using the secret key associated to $\mathsf{pk}_{\mathsf{cred}_\Delta}$ and σ_Δ is the signature of $(\{\mathsf{com}_{\Delta,i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\Delta})$ (part of cred_Δ) using $\mathsf{sk}_{\mathsf{Iss}}$.

We make the following observation.

Observation 2 *The challenger of the unforgeability experiment (Experiment 1) generates public keys* $\mathsf{pk}_{\mathsf{cred}}$ *and signs them (together with a set of commitments* $\{\mathsf{com}_i\}_{i\in[l]}$*) using* $\mathsf{sk}_{\mathsf{Iss}}$ *as a consequence of:*

1. *issuance queries: in this case, the challenger also gives to the adversary the associated secret key and adds the credential to* CT*;*
2. *delegation queries: in this case, the challenger also signs* pres' *in* π_{DP} *using the secret key* $\mathsf{sk}_{\mathsf{cred}}$ *associated to* $\mathsf{pk}_{\mathsf{cred}}$*, adds the delegation to* DT*, and does not reveal* $\mathsf{sk}_{\mathsf{cred}}$ *to the adversary.*
3. *delegated presentation queries: in this case, the issuer also signs* pres'' *in* π_{del} *using the secret key* $\mathsf{sk}_{\mathsf{cred}}$ *associated to* $\mathsf{pk}_{\mathsf{cred}}$*, adds the delegation to* PT*, and does not reveal* $\mathsf{sk}_{\mathsf{cred}}$ *to the adversary.*

Note that the challenger of Experiment 1 never signs public keys it has not generated .[16]

The adversary wins the experiment if $1 \leftarrow \mathsf{DelegPresVer}(\mathsf{pres}^\star)$ and at least one of the following conditions is satisfied:

1. *forgery of the delegation*: $\mathsf{del}^\star$ is forged, i.e.

[16] For the sake of clarity, note that in the security proof, the challenger of Experiment 1 is the reduction $\mathcal{B}$. The reduction does not know the secret key $\mathsf{sk}_{\mathsf{Iss}}$ but can ask for signatures of messages of its choice using $\mathsf{sk}_{\mathsf{Iss}}$ sending queries to $\mathcal{C}$, the challenger of the v-SUF-CMA experiment.

- it is not a delegation queried by $\mathcal{A}$ to $\mathsf{O}_{\mathsf{del}}$, i.e. $\mathsf{del}^\star \notin \mathsf{DT}$;
- $\mathsf{del}^\star$ was not generated using the credentials issued to $\mathcal{A}$ meaning that

Given $\mathsf{del}^\star = (\Delta_{\mathsf{ID}}{}^\star, \mathsf{scope}^\star, \mathsf{DP}^\star, \pi_{\mathsf{DP}^\star})$ with

$$\pi_{\mathsf{DP}^\star} = \left(\mathsf{pres}'^\star, \sigma'^\star = \mathsf{Sign}((\mathsf{pres}', \mathsf{DP}^\star, \mathsf{scope}^\star, \Delta_{\mathsf{ID}}{}^\star), \mathsf{sk}^\star_{\mathsf{cred}_\mathsf{D}})\right),$$

$$\mathsf{pres}'^\star = ((\sigma^\star_\mathsf{D}, \{\mathsf{com}^\star_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{pk}^\star_{\mathsf{cred}_\mathsf{D}}), \{\mathsf{salt}^\star_{\mathsf{D},i}\}_{i\in\mathsf{DP}}).$$

we say that $\mathsf{del}^\star$ was not generated using any credential issued to $\mathcal{A}$ if $\forall\, \mathsf{cred} \in \mathsf{CT},\ \mathsf{cred} = \big((\sigma_\mathsf{D}, \{\mathsf{com}_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\mathsf{D}}), \{a_{\mathsf{D},i}\}_{i\in[l]}, \{\mathsf{salt}_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{sk}_{\mathsf{cred}_\mathsf{D}}\big)$,

$$(\mathsf{pres}'^\star, \mathsf{DP}^{\star 17}) \neq \big((\sigma_\mathsf{D}, \{\mathsf{com}_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\mathsf{D}}), \{\mathsf{salt}_{\mathsf{D},i}\}_{i\in\mathsf{DP}^\star}, \{a_{\mathsf{D},i}\}_{i\in\mathsf{DP}^\star}\big),$$

because these are the information of the credential used to generate a delegation that appear in the delegation itself.

The public key $\mathsf{pk}^\star_{\mathsf{cred}_\mathsf{D}}$ in $\mathsf{del}^\star$ has been generated either by $\mathcal{A}$, by $\mathcal{B}$, or by $\mathcal{C}$.

$\mathsf{pk}^\star_{\mathsf{cred}_\mathsf{D}}$ is generated by $\mathcal{A}$. In this case none of the credentials issued to $\mathcal{A}$ is related to $\mathsf{pk}^\star_{\mathsf{cred}_\mathsf{D}}$. This means that the signature $\sigma^\star_\mathsf{D}$ of $(\{\mathsf{com}^\star_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\mathsf{D}})$ must be a forgery of $\mathsf{pk}_{\mathsf{Iss}}$ because $\mathcal{C}$ signs with $\mathsf{sk}_{\mathsf{Iss}}$ only public keys generated by $\mathcal{B}$ or $\mathcal{C}$ (according to our reduction definition);

$\mathsf{pk}^\star_{\mathsf{cred}_\mathsf{D}}$ is generated by $\mathcal{B}$. $\mathsf{pk}^\star_{\mathsf{cred}_\mathsf{D}}$ was included in a credential issued by $\mathsf{O}_{\mathsf{iss}}$, but the delegation $\mathsf{del}^\star$ is not generated using this credential. This means that one of the following events happens:

(a) the commitments $\{\mathsf{com}^\star_{\mathsf{D},i}\}_{i\in[l]} \neq \{\mathsf{com}_{\mathsf{D},i}\}_{i\in[l]} \vee \sigma^\star_D \neq \sigma_D$: in this case the adversary has broken the v-SUF-CMA of the $\mathsf{pk}_{\mathsf{Iss}}$;

(b) $\mathsf{DP}^\star \neq \{a_{\mathsf{D},i}\}_{i\in\mathsf{DP}^\star}$: in this case it is possible to reduce to the binding property of the commitment scheme (and hence to the collision resistance of H);

(c) $\{\mathsf{salt}^\star_{\mathsf{D},i}\}_{i\in\mathsf{DP}} \neq \{\mathsf{salt}_{\mathsf{D},i}\}_{i\in\mathsf{DP}}$ are different, and in this case it is possible to reduce to the collision resistance of H.

$\mathsf{pk}^\star_{\mathsf{cred}_\mathsf{D}}$ is generated by $\mathcal{C}$. $\mathsf{pk}^\star_{\mathsf{cred}_\mathsf{D}}$ has been generated by $\mathcal{C}$ during the generation of the $\mathsf{pk}_{\mathsf{Iss}}$, or to generate the response to queries to $\mathsf{O}_{\mathsf{del}}$ or $\mathsf{O}_{\mathsf{pres}}$. Then, neither $\mathcal{A}$ nor $\mathcal{B}$ know the secret counterpart $\mathsf{sk}_{\mathsf{cred}_\mathsf{D}}$. Since $\mathsf{del}^\star \notin \mathsf{DT}$, the challenger has never signed with $\mathsf{sk}^\star_{\mathsf{cred}_\mathsf{D}}$ the message $((\sigma_\mathsf{D}, \{\mathsf{com}_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\mathsf{D}}), \{\mathsf{salt}_{\mathsf{D},i}\}_{i\in\mathsf{DP}^\star}, \mathsf{DP}^\star, \mathsf{scope}^\star, \Delta_{\mathsf{ID}}{}^\star)$, therefore σ', included in $\pi_{\mathsf{DP}^\star}$, is a forgery of $\mathsf{pk}_{\mathsf{cred}_\mathsf{D}}$.

In this way we have shown that our reduction can reduce to the v-SUF-CMA of the signature scheme, to the binding property of the commitment scheme or to the collision resistance of the underlying hash function.

2. *forgery of the delegated presentation*: $\pi_{\mathsf{del}^\star}$ is forged, i.e.
   - it is not a presentation queried by $\mathcal{A}$, i.e. $\mathsf{pres}^\star \notin \mathsf{PT}$;
   - $\pi_{\mathsf{del}^\star}$ was not generated using the credentials issued to $\mathcal{A}$.

   Given

$$\pi_{\mathsf{del}^\star} = \left(\mathsf{pres}'', \sigma'' = \mathsf{Sign}((\mathsf{pres}'', , \mathsf{del}^\star), \mathsf{sk}_{\mathsf{cred}_\Delta})\right),$$

$$\mathsf{pres}'' = ((\sigma_\Delta, \{\mathsf{com}_{\Delta,i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\Delta}), \{\mathsf{salt}_{\Delta,i}\}_{i\in\Delta_{\mathsf{ID}}}, \mathsf{nonce}),$$

[17] Depending on the context, $\mathsf{DP}^\star$ denotes either the set of revealed attributes appearing in the delegation or the corresponding set of indices selecting those attributes.

we say $\pi_{\mathsf{del}^\star}$ was not generated using any credential issued to $\mathcal{A}$ if $\forall \mathsf{cred} \in \mathsf{CT}$,

$$\mathsf{cred} = \left((\sigma_\Delta, \{\mathsf{com}_{\Delta,i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\Delta}), \{a_{\Delta,i}\}_{i\in[l]}, \{\mathsf{salt}_{\Delta,i}\}_{i\in[l]}, \mathsf{sk}_{\mathsf{cred}_\Delta}\right),$$

defining $\overline{\mathsf{pres}''}$ removing $\mathsf{nonce}^\star$ from pres'',

$$\left(\overline{\mathsf{pres}''}, \Delta_{\mathsf{ID}}{}^\star\right) \neq \left((\sigma_\Delta, \{\mathsf{com}_{\Delta,i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_\Delta}), \{\mathsf{salt}_{\Delta,i}\}_{i\in\Delta_{\mathsf{ID}}{}^\star}, \{a_{\Delta,i}\}_{i\in\Delta_{\mathsf{ID}}{}^\star}\right).$$

Using a similar observation, it is possible to show that even in this case our reduction can reduce to the v-SUF-CMA of the underlying signature scheme or the binding property of the commitment scheme or the collision resistance of the hash function used to define the commitment.

This allows us to prove that an $\mathcal{A}$ winning the unforgeability of the delegation scheme experiment with non-negligible probability, can be used as a subroutine for a reduction winning one of the following experiments with non-negligible probability:

- the strong v-UF-CMA experiment of the underlying signature scheme or
- the collision resistance of the underlying hash function used to instantiate the random oracle.

means that at least one of the public keys that $\mathcal{B}$ has received by $\mathcal{C}$, corresponding to the sessions opened in the v-SUF-CMA experiment, has been forged. Therefore $\mathcal{B}$ win the v-SUF-CMA experiment, and this concludes the security proof.

Theorem 4. *The delegation scheme described in Definition 16 is unforgeable according to Definition 12 under the assumption that BBS is strongly unforgeable.* [18]

Since in this case the proofs included in the delegations and delegated presentations are NIZKPs bound to a specific context (for delegations $(\Delta_{\mathsf{ID}}, \mathsf{scope}, \mathsf{DP})$, whereas for delegated presentations a delegation del), we can leverage on the existence of a simulator that, by programming the random oracle can generate proofs that are indistinguishable from real proofs. Therefore, in this case, there is not need for the reduction to generate the credentials used to generate the proofs π_{DP} and π_{del}. This remarkably simplifies the security analysis making it more linear, and enabling the design of a reduction to the unforgeability of the BBS signature scheme.

Proof Sketch. We prove that if there exists an adversary $\mathcal{A}$ of Experiment 1 who can win the experiment with non-negligible probability, then we can use $\mathcal{A}$ to build a reduction $\mathcal{B}$ to the SUF-CMA experiment of the BBS signature scheme, which wins the experiment with non-negligible probability. The reduction $\mathcal{B}$ interacts with the challenger $\mathcal{C}$ of the SUF-CMA experiment and simulates the challenger of Experiment 1.

Setup $\mathcal{B}$ receives the public parameters pp and the public key $\mathsf{pk}_{\mathsf{Iss}}$ of the BBS signature scheme from the challenger $\mathcal{C}$ of the SUF-CMA experiment. $\mathcal{B}$ forwards $(\mathsf{pp}, \mathsf{pk}_{\mathsf{Iss}})$ to the adversary $\mathcal{A}$.

Training phase We show how $\mathcal{B}$ answers to the queries $\mathcal{A}$ can send during the training phase.

[18] Which holds under q-Strong Diffie-Hellman assumption [43]. For further details, see Appendix .3.

- Queries to $\mathsf{O}_{\mathsf{iss}}$: $\mathcal{A}$ sends in input a set of attributes $\{a_i\}_{i\in[l]}$. $\mathcal{B}$ performs the following operations:
  1. it sends a signing query to $\mathcal{C}$ with input $\{a_i\}_{i\in[l]}$: $\mathcal{C}$ returns a signature σ of $\{a_i\}_{i\in[l]}$ generated using $\mathsf{sk}_{\mathsf{Iss}}$.
  2. $\mathcal{B}$ sends to $\mathcal{A}$ the credential $\mathsf{cred} \leftarrow (\sigma, \{a_i\}_{i\in[l]})$ and adds cred to CT.
- Queries to $\mathsf{O}_{\mathsf{del}}$: $\mathcal{A}$ can query for a delegation for the values $(\varDelta_{\mathsf{ID}}, \mathsf{scope}, \mathsf{DP})$ of its choice. Upon receiving a query $\mathcal{B}$ performs the following operations:
  1. runs the simulator, generating a proof $\pi_{\mathsf{DP}} = (\bar{A}_{\mathsf{D}}, \bar{B}_{\mathsf{D}}, c_1, z_1)$ for the statement DP by programming the random oracle H on the input $(\bar{A}_{\mathsf{D}}, \bar{B}_{\mathsf{D}}, U_{\mathsf{D}}, \mathsf{DP}, \mathsf{scope}, \varDelta_{\mathsf{ID}})$. This operation overwrites the random oracle only with negligible probability because the values $\bar{A}_{\mathsf{D}}, \bar{B}_{\mathsf{D}}, U_{\mathsf{D}}$ generated by the simulator have super-logarithmic min-entropy [43].
  2. $\mathcal{B}$ sends to $\mathcal{A}$ the delegation $\mathsf{del} = (\varDelta_{\mathsf{ID}}, \mathsf{scope}, \mathsf{DP}, \pi_{\mathsf{DP}})$ and adds del to DT.
- Queries to $\mathsf{O}_{\mathsf{pres}}$: $\mathcal{A}$ can query for a delegated presentation giving in input $(\mathsf{del}, \mathsf{nonce})$ of its choice. Upon receiving a query $\mathcal{B}$ performs the following operations:
  1. runs the simulator, generating a proof $\pi_{\mathsf{del}} = (\bar{A}_{\varDelta}, \bar{B}_{\varDelta}, c_2, z_2)$ for the statement $\varDelta_{\mathsf{ID}}$ by programming the random oracle H on the input $(\bar{A}_{\varDelta}, \bar{B}_{\varDelta}, U_{\varDelta}, \mathsf{del})$. This operation overwrites the random oracle only with negligible probability because the values $\bar{A}_{\varDelta}, \bar{B}_{\varDelta}, U_{\varDelta}$ generated by the simulator have super-logarithmic min-entropy [43].
  2. $\mathcal{B}$ sends to $\mathcal{A}$ the presentation $\mathsf{pres} = (\mathsf{del}, \pi_{\mathsf{del}})$ and adds pres to PT.

Forgery Phase $\mathcal{A}$ sends to $\mathcal{B}$ a delegated presentation $\mathsf{pres}^\star = (\mathsf{del}^\star, \pi_{\mathsf{del}^\star})$ as its forgery. To be a valid forgery, this presentation must satisfy at least one of the winning conditions of Experiment 1.

Instantiation and analysis of the winning condition of Experiment 1 Note that, since we are describing a VP delegation scheme that is built on top of an anonymous credential scheme, the proofs generated by delegator and delegatee hide the anonymous credential used to generate it. Therefore, in this case the statement "$\mathsf{del}^\star$ (or $\pi_{\mathsf{del}^\star}$) is not generated using any credential issued to $\mathcal{A}$" means that none of the issued credentials can be used to generate the associated proof π_{DP} (or indeed $\pi_{\mathsf{del}^\star}$) included in $\mathsf{del}^\star$ (or in $\mathsf{pres}^\star$), i.e. none of the credentials issued to the adversary satisfy the statements, or set of attributes in DP (or $\varDelta_{\mathsf{ID}}$).

We consider separately the two cases:

1. $\mathsf{del}^\star$ *is forged, i.e.* $\mathsf{del}^\star \notin \mathsf{DT}$ *and* $\forall \mathsf{cred} \in \mathsf{CT}$*, being* $\mathbf{A}$ *the set of attributes in* cred*, then* $\mathsf{DP}^\star \not\subseteq \mathbf{A}$.
   Since the proof $\pi_{\mathsf{DP}^\star}$ is a NIZKP obtained by applying the Fiat-Shamir transform to a sigma protocol, it is possible to run the Fiat-Shamir extractor and extract a BBS signature. This is done by rewinding $\mathcal{A}$ to the moment it has sent the random oracle query associated to the challenge associated to the proof $\pi_{\mathsf{DP}^\star}$, and changing the associated digest. With non-negligible probability $\mathcal{A}$ will produce another valid forgery returning two transcript of the sigma protocol for the same first message but two different challenges. From this $\mathcal{B}$ can extract a witness, i.e. a BBS signature $\sigma^\star$, for the statement DP. But the adversary was never issuer a BBS signature for the attributes in DP and this means that $\mathcal{B}$ has never received from $\mathcal{C}$ the signature $\sigma^\star$ and can win the SUF-CMA experiment for BBS signatures by sending $\sigma^\star$ to $\mathcal{C}$.
2. $\pi_{\mathsf{del}^\star}$ *is forged,* $\forall \mathsf{pres} \in \mathsf{PT}$*,* $\mathsf{pres} = (\mathsf{del}, \pi_{\mathsf{del}})$*,* $\pi_{\mathsf{del}} \neq \pi_{\mathsf{del}^\star}$ *and* $\forall \mathsf{cred} \in \mathsf{CT}$*, being* $\mathbf{A}$ *the set of attributes in* cred*, then* ${\varDelta_{\mathsf{ID}}}^\star \not\subseteq \mathbf{A}$. This follows from a similar argument as the previous item. It is possible to show that the reduction can extract a BBS

signature $\sigma^\star$ for the attributes in Δ_{ID} that were never issued to $\mathcal{A}$, and therefore it was not generated by $\mathcal{C}$. This means that $\mathcal{B}$ can win the SUF-CMA experiment for BBS signatures by returning to $\mathcal{C}$ the signature $\sigma^\star$.

6.3 Unlinkability

In this section, we focus on proving the *unlinkability* of delegated presentations for the delegation scheme described in Definition 16. We note that the unlinkability of the delegation issuance proof follows from the same argument, since the interactive proof used in $\mathsf{DelegIssuance}$ employs the same building blocks as the one used in $\mathsf{DelegPres}$.

Intuitively, the proof is presented showing the existence of a probabilistic polynomial-time simulator $\mathcal{S}$ that can generate a simulated presentation indistinguishable from that produced by an honest delegatee holding a valid BBS credential. In fact, if such a simulator exists, this implies that no adversary can distinguish between two honest delegated presentations derived from different credentials, as long as they are consistent with the same Δ_{ID}. We formalize this argument by explicitly describing an indistinguishability experiment and the associated security definition.

This explicit formalization is useful to relate two kinds of definition of unlinkability that are used in the literature: one stronger, which requires the existence of a simulator that generates presentations that are indistinguishable from honestly generated ones, and a weaker one where the adversary generates and sends to the challenger two credentials and it must guess which of the two credentials was used to generate a presentation it receives from the challenger.

Indistinguishability Experiment and Definition. The goal of this experiment is to ensure that the delegated presentation generated using a real anonymous credential is indistinguishable from one generated by a simulator $\mathcal{S}$ that does not possess the credential. The existence of such a simulator implies that the delegated presentation does not reveal any information about the underlying credential. In particular, we will show that if no efficient adversary can win the indistinguishability experiment with non-negligible advantage, then no adversary can win the unlinkability experiment with non-negligible advantage — that is, no adversary can distinguish which of two credentials was used to produce a given delegated presentation with probability significantly greater than $\frac{1}{2}$.

Experiment 4 ($\mathsf{Exp}_{\mathcal{A}}^{\mathsf{Indistinguishability}}(1^\lambda)$)

1. *The adversary $\mathcal{A}$ generates the public parameters* pp *for the verifiable credential scheme together with the issuer key pair* $(\mathsf{sk}_{\mathsf{Iss}}, \mathsf{pk}_{\mathsf{Iss}})$ *and send* pp *and* $\mathsf{pk}_{\mathsf{Iss}}$ *to the challenger.*
2. *The challenger samples a random bit* $b' \xleftarrow{\$} \{0,1\}$;
3. *The adversary $\mathcal{A}$ generates and sends to the challenger* del, nonce *and* cred_Δ *satisfying* Δ_{ID}; [19]
4. *The challenger computes* pres *as follows:*
   - *If* $b' = 0$, *computes* $\mathsf{pres} \xleftarrow{\$} \mathcal{S}(\mathsf{del}, \mathsf{nonce})$
   - *If* $b' = 1$, *computes* $\mathsf{pres} \xleftarrow{\$} \mathsf{DelegPres}(\mathsf{del}, \mathsf{cred}_\Delta, \mathsf{nonce})$.

[19] The challenger verifies that: del is valid, and that cred_Δ satisfies Δ_{ID} and is valid under $\mathsf{pk}_{\mathsf{Iss}}$. The verification algorithm for a VC is not defined in this work, however every VC scheme that we consider has a natural associated algorithm to verify the correct issuance of the credential.

5. *The challenger sends* pres *to* $\mathcal{A}$.
6. $\mathcal{A}$ *outputs a bit* $\bar{b}'$.
7. $\mathcal{A}$ *wins if* $\bar{b}' = b'$.

If the adversary has access to a random oracle (according to the VP delegation scheme definition), it can send random oracle queries before sending to the challenger del, nonce *and* cred_{Δ}. *In that case, if* $b' = 0$ *the random oracle will be simulated by* $\mathcal{S}$, *otherwise the queries are sent to a real random oracle.*

Definition 17 (Indistinguishability of Delegated Presentations). *A VP delegation scheme satisfies* indistinguishability of the delegated presentation *if for any probabilistic polynomial-time (PPT) adversary* $\mathcal{A}$ *executing the experiment* $\mathsf{Exp}_{\mathcal{A}}^{\mathsf{Indistinguishability}}(1^{\lambda})$,

$$\left|\Pr\left[\mathcal{A} \textit{ wins } \mathsf{Exp}_{\mathcal{A}}^{\mathsf{Indistinguishability}}(1^{\lambda})\right] - \frac{1}{2}\right| = \left|\Pr[\bar{b}' = b'] - \frac{1}{2}\right| \leq \nu(\lambda),$$

where $\nu(\lambda)$ *is a negligible function in the security parameter* λ.

We now prove that our VP delegation scheme built on top of the BBS AC scheme satisfies the security notion of indistinguishability of delegated presentations.

Theorem 5. *The delegation scheme described in Definition 16 satisfies* indistinguishability of the delegated presentation *as defined in Definition 17.*

Proof Sketch. We construct a simulator $\mathcal{S}$ that emulates the delegated presentation by programming the random oracle RO.

- **Input.** The simulator $\mathcal{S}$ takes as input the public parameters of the system, the issuer's public key $\mathsf{pk}_{\mathsf{Iss}}$, and a pair $(\bar{A}_c, \bar{B}_c)$ that satisfies the relation $\bar{A}_c^{\mathsf{sk}_{\mathsf{Iss}}} = \bar{B}_c$ [14,43].
- **Random Oracle queries.** At the beginning, the internal hash table HT is empty. Whenever a query is made to the random oracle RO:
  - If the input has already been queried, the simulator returns the corresponding value stored in the hash table HT;
  - If the input x is new, the simulator samples a random value $c \xleftarrow{\$} \mathbb{Z}_p$, stores (x, c) in HT, and returns x.

The simulator $\mathcal{S}$ proceeds as follows:

- Sample a random exponent $\gamma \xleftarrow{\$} \mathbb{Z}_p$.
- Compute:
$$\bar{A} \leftarrow \bar{A}_c^{\gamma}, \qquad \bar{B} \leftarrow \bar{B}_c^{\gamma}.$$
- Sample random values:
$$s, n \xleftarrow{\$} \mathbb{Z}_p, \qquad u_i \xleftarrow{\$} \mathbb{Z}_p \quad \forall i \in \mathsf{Hid}.$$
- Set the randomness vector:
$$\mathbf{z} \leftarrow (s, n, (u_i)_{i \in \mathsf{Hid}}).$$
- Sample a random challenge:
$$c \xleftarrow{\$} \mathbb{Z}_p.$$

– Compute:

$$U \leftarrow \bar{A}^s \cdot \bar{B}^n \cdot \prod_{i \in \mathsf{Hid}} \mathbf{h}[i]^{u_i} \cdot C_{\mathsf{Rev}}(\mathbf{a})^{-c}.$$

The simulator $\mathcal{S}$ then programs the random oracle RO on the tuple $(\bar{A}, \bar{B}, U, \mathsf{del}, \mathsf{nonce})$, by storing $((\bar{A}, \bar{B}, U, \mathsf{del}, \mathsf{nonce}), c)$ in HT. This ensures that any future query with the same input will consistently return the fixed challenge c. In Appendix .4 we analyze the conditions under which the simulation fails. We compute the probability of this event and show that it is negligible in the security parameter λ.

The simulator $\mathcal{S}$ then constructs the proof and the corresponding delegated presentation as follows:

– It sets the proof π_{del} to:

$$\pi_{\mathsf{del}} \leftarrow (\bar{A}, \bar{B}, c, \mathbf{z});$$

– It assembles the final presentation pres as:

$$\mathsf{pres} \leftarrow (\mathsf{del}, \pi_{\mathsf{del}})$$

and sends it to the adversary.

Note that the simulator $\mathcal{S}$ is constructed to ensure that the output presentations are correct and distributed as real presentations.

In particular,

$$\bar{A}^{\mathsf{sk}_{\mathsf{Iss}}} = (\bar{A}_c^{\gamma})^{\mathsf{sk}_{\mathsf{Iss}}} = (\bar{A}_c^{\mathsf{sk}_{\mathsf{Iss}}})^{\gamma} = \bar{B}_c^{\gamma} = \bar{B}$$

therefore, the pairing check

$$\mathbf{e}(\bar{A}, X_2) = \mathbf{e}(\bar{B}, g_2)$$

remains satisfied after randomization, and the pair $(\bar{A}, \bar{B})$ is distributed uniformly at random as in real presentations. In addition, the first message U is chosen uniformly at random as a consequence of the sampling of the responses $\mathbf{z}$ and the challenge c

$$U = \bar{A}^s \cdot \bar{B}^n \cdot \prod_{i \in \mathsf{Hid}} \mathbf{h}[i]^{u_i} \cdot C_{\mathsf{Rev}}(\mathbf{a})^{-c}.$$

and since the simulator programs the random oracle on input $(\bar{A}, \bar{B}, U, \mathsf{del}, \mathsf{nonce})$ to c, the verification still holds. Consequently, all verification checks are satisfied, and the simulated presentation is indistinguishable from a real one. This means that, in Experiment 4, an adversary trying to distinguish a simulated presentation (obtained running $\mathcal{S}$) from a real one (obtained running $\mathsf{DelegPres}$) cannot succeed with probability greater than $\frac{1}{2}$. Therefore, it satisfies the indistinguishability of delegated presentations as defined in Definition 17.

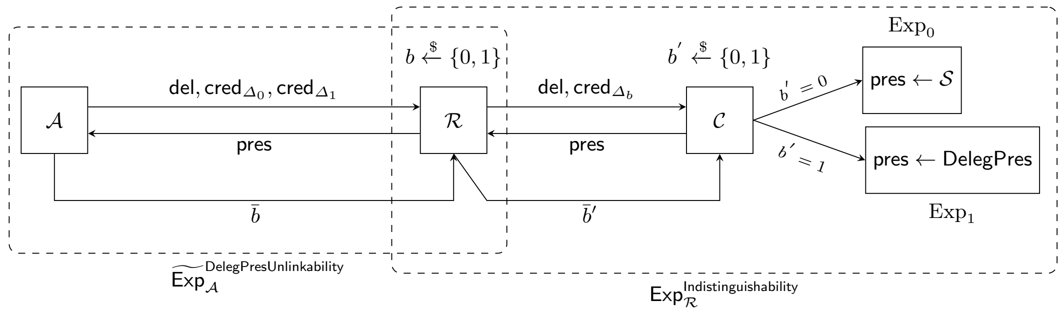
Now we show that a VP delegation scheme that satisfies the indistinguishability of delegated presentation (Definition 17 also satisfies the unlinkability of delegated presentation (Definition 14). The same clearly holds for the unlinkability of delegation issuance, even if it is not explicitly described.

Theorem 6. *The delegation scheme described in Definition 16 satisfies unlinkability*[20] *of the delegated presentation as defined in Definition 14.*

We prove the theorem via a reduction. Specifically, we assume that there exists a probabilistic polynomial-time adversary $\mathcal{A}$ that wins the unlinkability experiment $\mathsf{Exp}_{\mathcal{A}}^{\mathsf{DelegPresUnlinkability}}(1^\lambda)$ with non-negligible probability. We then construct a reduction $\mathcal{R}$ that uses $\mathcal{A}$ to win the indistinguishability experiment $\mathsf{Exp}_{\mathcal{R}}^{\mathsf{Indistinguishability}}(1^\lambda)$ with non-negligible probability. This would contradict the result proven in Theorem 5.

Proof Sketch. Consider the probabilistic polynomial-time simulator $\mathcal{S}$ introduced in the proof of Theorem 5, which, given as input del and nonce, outputs a simulated delegated presentation pres^* that is computationally indistinguishable from a real delegated presentation pres generated by an honest delegatee using a valid anonymous credential $\mathsf{anoncred}_{\Delta}$, i.e.,

$$\mathcal{S}(\mathsf{del}, \mathsf{nonce}) \to \mathsf{pres}^* \approx_c \mathsf{DelegPres}(\mathsf{del}, \mathsf{cred}_{\Delta}, \mathsf{nonce}).$$



$b \xleftarrow{\$} \{0,1\}$
$b' \xleftarrow{\$} \{0,1\}$
Exp$_0$
$\mathsf{del}, \mathsf{cred}_{\Delta_0}, \mathsf{cred}_{\Delta_1}$
$\mathsf{del}, \mathsf{cred}_{\Delta_b}$
$b' = 0$
$\mathsf{pres} \leftarrow \mathcal{S}$
$\mathcal{A}$
$\mathcal{R}$
$\mathcal{C}$
pres
pres
$b' = 1$
$\mathsf{pres} \leftarrow \mathsf{DelegPres}$
$\bar{b}$
$\bar{b}'$
Exp$_1$
$\widetilde{\mathsf{Exp}}_{\mathcal{A}}^{\mathsf{DelegPresUnlinkability}}$
$\mathsf{Exp}_{\mathcal{R}}^{\mathsf{Indistinguishability}}$

Fig. 3. Overview of the interaction between the adversary $\mathcal{A}$, the challenger $\mathcal{C}$, and the reduction $\mathcal{R}$. The reduction $\mathcal{R}$ first chooses a bit b to select one of the two credentials provided by $\mathcal{A}$. It then receives either a simulated presentation (if $b' = 0$) or a real presentation (if $b' = 1$) corresponding to the chosen credential. The adversary $\mathcal{A}$ tries to guess the bit b based on the presentation received. Based on $\mathcal{A}$'s guess $\bar{b}$, the reduction $\mathcal{R}$ outputs its own guess $\bar{b'}$ for b': if $\bar{b} = b$, $\mathcal{R}$ guesses that the presentation is real ($b' = 1$); otherwise, it guesses simulated ($b' = 0$).

Assume, by contradiction, that there exists an adversary $\mathcal{A}$ that wins $\mathsf{Exp}_{\mathcal{A}}^{\mathsf{DelegPresUnlinkability}}(1^\lambda)$ with probability $\frac{1}{2} + \epsilon$, where ϵ is non-negligible in the security parameter λ. Recall from Experiment 3 that this corresponds to the setting where the presentation is always generated from a real credential.

[20] In the delegation scheme described in Definition 15, unlinkability does not hold for delegations and delegated presentations. Specifically, the delegator's and delegate's presentations are structured as

$$\mathsf{pres}' \leftarrow \big((\sigma_{\mathsf{D}}, \{\mathsf{com}_{\mathsf{D},i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_{\mathsf{D}}}), \{\mathsf{salt}_{\mathsf{D},i}\}_{i\in\mathsf{DP}}\big),$$
$$\mathsf{pres}'' \leftarrow \big((\sigma_{\Delta}, \{\mathsf{com}_{\Delta,i}\}_{i\in[l]}, \mathsf{pk}_{\mathsf{cred}_{\Delta}}), \{\mathsf{salt}_{\Delta,i}\}_{i\in\Delta_{\mathsf{ID}}}, \mathsf{nonce}\big).$$

Since pres' and pres'' include $\mathsf{pk}_{\mathsf{cred}_{\mathsf{D}}}$ and $\mathsf{pk}_{\mathsf{cred}_{\Delta}}$, respectively, multiple presentations by the same party are trivially linkable.

We now introduce a variant of the Experiment 3, denoted by $\widetilde{\mathsf{Exp}}_{\mathcal{A}}^{\mathsf{DelegPresUnlinkability}}$, in which the presentation pres may be either real or simulated depending on a random bit b', as illustrated in Figure 3. The formal description of this variant is given in the following experiment.

Experiment 5 ($\widetilde{\mathsf{Exp}}_{\mathcal{A}}^{\mathsf{DelegPresUnlinkability}}(1^\lambda)$)

1. *The adversary $\mathcal{A}$ generates the public parameters pp for the verifiable credential scheme together with the issuer key pair $(\mathsf{sk}_{\mathsf{Iss}}, \mathsf{pk}_{\mathsf{Iss}})$ and send pp and $\mathsf{pk}_{\mathsf{Iss}}$ to the challenger.*
2. *The challenger samples a random bit $b' \xleftarrow{\$} \{0,1\}$.*
3. *$\mathcal{A}$ generates and sends to the challenger del, nonce and cred_{Δ_0} and cred_{Δ_1}, satisfying Δ_{ID}.*
4. *$\mathcal{C}$ samples a random bit $b' \xleftarrow{\$} \{0,1\}$ and computes pres as follows:*
   - *If $b' = 0$, computes $\mathsf{pres} \xleftarrow{\$} \mathcal{S}(\mathsf{del}, \mathsf{nonce})$.*
   - *If $b' = 1$, samples a random bit b and computes $\mathsf{pres} \xleftarrow{\$} \mathsf{DelegPres}(\mathsf{del}, \mathsf{cred}_{\Delta_b}, \mathsf{nonce})$.*
5. *$\mathcal{C}$ sends pres to $\mathcal{A}$.*
6. *$\mathcal{A}$ outputs a bit $\bar{b}$.*
7. *$\mathcal{A}$ wins if $\bar{b} = b$.*

If the adversary has access to a random oracle (according to the VP delegation scheme definition), it can send random oracle queries before sending to the challenger $\mathsf{del}, \mathsf{nonce}$, cred_{Δ_0} and cred_{Δ_1}.

Next, we proceed to compute the probability that the adversary $\mathcal{A}$ wins $\widetilde{\mathsf{Exp}}_{\mathcal{A}}^{\mathsf{DelegPresUnlinkability}}$:

$$
\begin{aligned}
\Pr\left[\mathcal{A} \text{ wins } \widetilde{\mathsf{Exp}}_{\mathcal{A}}^{\mathsf{DelegPresUnlinkability}}\right] &= \Pr\left[\bar{b} = b\right] = \\
&= \Pr\left[\bar{b} = b \mid b' = 0\right] \cdot \Pr\left[b' = 0\right] \\
&\quad + \Pr\left[\bar{b} = b \mid b' = 1\right] \cdot \Pr\left[b' = 1\right] = \\
&= \Pr\left[\bar{b} = b \mid b' = 0\right] \cdot \tfrac{1}{2} \\
&\quad + \Pr\left[\mathcal{A} \text{ wins } \mathsf{Exp}_{\mathcal{A}}^{\mathsf{DelegPresUnlinkability}}\right] \cdot \tfrac{1}{2} = \\
&= \tfrac{1}{2} \cdot \tfrac{1}{2} + \left(\tfrac{1}{2} + \epsilon\right) \cdot \tfrac{1}{2} \\
&= \tfrac{1}{2} + \tfrac{\epsilon}{2}.
\end{aligned}
$$

This reflects the fact that adversary $\mathcal{A}$ has an advantage ϵ only when $b' = 1$, since otherwise it guesses at random.

Next, consider the probability that the bit $b' = 1$ conditioned on the event that $\mathcal{A}$ guesses correctly, i.e., $\bar{b} = b$. By Bayes' theorem, we have

$$
\begin{aligned}
\Pr\left[b' = 1 \mid \bar{b} = b\right] &= \frac{\Pr\left[\bar{b} = b \mid b' = 1\right] \Pr[b' = 1]}{\Pr\left[\bar{b} = b\right]} = \\
&= \frac{\left(\frac{1}{2} + \epsilon\right) \cdot \frac{1}{2}}{\frac{1}{2} + \frac{\epsilon}{2}} = \frac{\frac{1}{2} + \epsilon}{1 + \epsilon} = \frac{1}{2} \cdot \frac{1 + 2\epsilon}{1 + \epsilon} = \frac{1}{2}\left(1 + \frac{\epsilon}{1 + \epsilon}\right) = \frac{1}{2} + \frac{1}{2} \cdot \frac{\epsilon}{1 + \epsilon}.
\end{aligned}
$$

Since ϵ is non-negligible and positive, and since

$$\frac{\epsilon}{1+\epsilon} > \frac{\epsilon}{2} \qquad \forall \;\; 0 < \epsilon \leq \frac{1}{2},$$

we obtain that

$$\Pr\left[b' = 1 \mid \bar{b} = b\right] > \frac{1}{2} + \frac{\epsilon}{4},$$

which is strictly greater than $\frac{1}{2}$ by a non-negligible quantity.

Consequently, $\mathcal{R}$ can define its own guess for b' as

$$\bar{b}' = \begin{cases} 1 & \text{if } \bar{b} = b, \\ 0 & \text{if } \bar{b} \neq b. \end{cases}$$

Finally, we compute the probability that $\mathcal{R}$ wins $\mathsf{Exp}_{\mathcal{R}}^{\mathsf{Indistinguishability}}$:

$$\begin{aligned}
\Pr\left[\mathcal{R} \text{ wins } \mathsf{Exp}_{\mathcal{R}}^{\mathsf{Indistinguishability}}\right] &= \Pr\left[\bar{b}' = b'\right] = \\
&= \Pr\left[\bar{b}' = 0 \mid b' = 0\right] \cdot \Pr\left[b' = 0\right] \\
&\quad + \Pr\left[\bar{b}' = 1 \mid b' = 1\right] \cdot \Pr\left[b' = 1\right] = \\
&= \Pr\left[\bar{b} \neq b \mid b' = 0\right] \cdot \tfrac{1}{2} \\
&\quad + \Pr\left[\bar{b} = b \mid b' = 1\right] \cdot \tfrac{1}{2} = \\
&= \Pr\left[\bar{b} \neq b \mid b' = 0\right] \cdot \tfrac{1}{2} \\
&\quad + \Pr\left[\mathcal{A} \text{ wins } \mathsf{Exp}_{\mathcal{A}}^{\mathsf{DelegPresUnlinkability}}\right] \cdot \tfrac{1}{2} \\
&= \tfrac{1}{2} \cdot \tfrac{1}{2} + \left(\tfrac{1}{2} + \epsilon\right) \cdot \tfrac{1}{2} = \\
&= \tfrac{1}{2} + \tfrac{\epsilon}{2}.
\end{aligned}$$

Therefore, any non-negligible advantage ϵ that $\mathcal{A}$ has in winning $\mathsf{Exp}_{\mathcal{A}}^{\mathsf{DelegPresUnlinkability}}$ implies an advantage of $\frac{\epsilon}{2}$ for an adversary $\mathcal{R}$ in winning $\mathsf{Exp}_{\mathcal{R}}^{\mathsf{Indistinguishability}}$. This contradicts the fact that, as proven in Theorem 5, the simulator $\mathcal{S}$ produces presentations that are computationally indistinguishable from real ones. We therefore conclude that our VP delegation scheme achieves *unlinkability* of delegated presentations.

Analogously, the unlinkability of the delegation issuance phase can be established using the same proof strategy. This is because the underlying zero-knowledge protocol used during delegation issuance is structurally identical to the one employed in delegated presentations, allowing for a simulator to generate indistinguishable transcripts.

7 Instantiation of Our VP Delegation Scheme for mdoc VCs in the EUDI and EBSI Frameworks

In this section we focus on the VP delegation scheme described in Definition 15, since mdoc VCs are the credentials supported by the EUDI ARF at the time of writing. The VP delegation scheme for mdoc VCs that we have described and analyzed can be integrated into existing ecosystems such as the EUDI or EBSI frameworks, without defining new data structures, only new verification procedures. We sketch some considerations describing possible solutions for that.

The delegation del can be a *VC issued by the delegator* that has as attributes the components we have described, namely $\mathsf{scope}, \Delta_{\mathsf{ID}}, \mathsf{DP}$ and π_{DP}. The delegator signs this credential (as an issuer) using the same secret key used to generate the π_{DP} as specified in Definition 15. This additional signature on the whole delegation is needed for compatibility reasons, because every VC must be signed by the issuer, in this case the delegator. This special credential must also contain the $\mathsf{pk}_{\mathsf{cred}}$ of the credential that the delegatee will use to present the delegation. In this sense, the delegatee must choose in advance the single-use credential it will use to generate the presentation.

When the delegatee creates a delegated presentation, it presents del (which is structured as a VC) disclosing all its attributes, namely, $\mathsf{scope}, \Delta_{\mathsf{ID}}, \mathsf{DP}$ and π_{DP}; the delegatee then creates a verifiable presentation π_{del} using its own VC, associated to $\mathsf{pk}_{\mathsf{cred}}$ communicated to the delegator, revealing the attributes included in Δ_{ID}. The outcome of the delegated presentation is derived from the combination of these two presentations. The only modification to the verification protocol is that the verifier must check that π_{DP} is indeed a valid presentation of the statement DP and that the presentation π_{del} created by the delegatee is a valid presentation of Δ_{ID}.

In EBSI, the only entities entitled to issue credentials are legal persons whose DID (a reference to, at least, their public key) is registered in the Trusted Issuer Registry (TIR)[21]. This means that credential holders who are physical persons are not allowed to issue credentials and therefore cannot generate the delegation as we have mentioned above. If the delegator is only a physical person we must consider two cases:

- the delegatee is a legal person: in this case, the delegator can generate the attributes for the delegation VC $\mathsf{scope}, \Delta_{\mathsf{ID}}, \mathsf{DP}$ and π_{DP} and sends them to the delegatee who generates the VC containing this information and signs it in turn. Note that this signature is only useful to guarantee the compatibility with the credential format and to create a VC issued by an entity registered in the TIR. The delegatee can choose not to sign it, which is perfectly fine, but cannot change the information sent by the delegator because π_{DP} is cryptographically bound to $\mathsf{scope}, \Delta_{\mathsf{ID}}$ and DP.
- if the delegatee is also only a physical person, the delegator must have the delegation VC signed by a third party registered in the TIR, which may be the issuer of the credential used to generate the delegation or a third party with this specific role.

Acknowledgments. We thank Giada Sciarretta for the discussion about the application of our protocol in the EUDI framework. Andrea Gangemi is a member of GNSAGA of INdAM. Andrea Flamini, Andrea Gangemi and Enrico Guglielmino are also members of CrypTO, the group of Cryptography and Number Theory of the Politecnico di Torino. Andrea Gangemi and Andrea Flamini are supported by the QUBIP project, funded by the European Union under the Horizon Europe framework program [grant agreement no. 101119746]. Vincenzo Orabona is Blockchain Specialist for Eustema S.p.A and works on large-scale pilots developed by EBSI.

[21] `https://hub.ebsi.eu/apis/pilot/trusted-issuers-registry/v4`

References

1. The European Digital Identity Wallet Architecture and Reference Framework, version 2.6.0 (10 2025), https://eu-digital-identity-wallet.github.io/eudi-doc-architecture-and-reference-framework/2.6.0/
2. The European Digital Identity Wallet Reference Implementation Roadmap. https://github.com/orgs/eu-digital-identity-wallet/projects/24/views/2 (10 2025), https://github.com/orgs/eu-digital-identity-wallet/projects/24/views/2
3. Mobile Driver's License (mDL) Implementation Guidelines, Version 1.2. https://www.aamva.org/topics/mobile-driver-license (01 2023), https://www.aamva.org/topics/mobile-driver-license
4. Abdalla, M., An, J.H., Bellare, M., Namprempre, C.: From identification to signatures via the fiat-shamir transform: Minimizing assumptions for security and forward-security. In: International conference on the theory and applications of cryptographic techniques. pp. 418–433. Springer (2002)
5. Au, M.H., Susilo, W., Mu, Y.: Constant-size dynamic k-TAA. Cryptology ePrint Archive, Paper 2008/136 (2008), https://eprint.iacr.org/2008/136
6. Barreto, P.S., Lynn, B., Scott, M.: Constructing elliptic curves with prescribed embedding degrees. In: Security in Communication Networks: Third International Conference, SCN 2002 Amalfi, Italy, September 11–13, 2002 Revised Papers 3. pp. 257–267. Springer (2003)
7. Baum, C., Blazy, O., Camenisch, J., Hoepman, J.H., Lee, E., Lehmann, A., Lysyanskaya, A., Mayrhofer, R., Montgomery, H., Nguyen, N.K., et al.: Cryptographers' Feedback on the EU Digital Identity's ARF. Tech. Rep. (2024)
8. Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H.: Randomizable proofs and delegatable anonymous credentials. In: Advances in Cryptology-CRYPTO 2009: 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings. pp. 108–125. Springer (2009)
9. Blömer, J., Bobolz, J.: Delegatable attribute-based anonymous credentials from dynamically malleable signatures. In: International Conference on Applied Cryptography and Network Security. pp. 221–239. Springer (2018)
10. Boneh, D., Boyen, X.: Short signatures without random oracles and the SDH assumption in bilinear groups. Journal of Cryptology **21**(2), 149–177 (2008), available at http://www.cs.stanford.edu/~xb/joc07/
11. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: CRYPTO 2004. LNCS, vol. 3152, pp. 41–55 (08 2004). https://doi.org/10.1007/978-3-540-28628-8_3, https://doi.org/10.1007/978-3-540-28628-8_3
12. Boneh, D., Shoup, V.: A Graduate Course in Applied Cryptography (2023), https://toc.cryptobook.us/book.pdf, last accessed: 2025-10-10
13. Camenisch, J., Drijvers, M., Dubovitskaya, M.: Practical UC-secure delegatable credentials with attributes and their application to blockchain. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. pp. 683–699 (2017)
14. Camenisch, J., Drijvers, M., Lehmann, A.: Anonymous attestation using the strong Diffie Hellman assumption revisited. In: Trust 2016. LNCS, vol. 9824, pp. 1–20 (2016). https://doi.org/10.1007/978-3-319-45572-3_1
15. Camenisch, J., Lysyanskaya, A.: A signature scheme with efficient protocols. In: SCN 2002. LNCS, vol. 2576, pp. 268–289 (2002). https://doi.org/10.1007/3-540-36413-7_20
16. Camenisch, J., Lysyanskaya, A.: Signature schemes and anonymous credentials from bilinear maps. In: Annual international cryptology conference. pp. 56–72. Springer (2004)
17. Chairattana-Apirom, R., Harding, F., Lysyanskaya, A., Tessaro, S.: Server-aided anonymous credentials. In: Annual International Cryptology Conference. pp. 291–324. Springer (2025)
18. Chairattana-Apirom, R., Hofheinz, D., Tessaro, S.: Tight security for bbs signatures. Cryptology ePrint Archive (2025)

19. Chairattana-Apirom, R., Tessaro, S.: On the concrete security of bbs/bbs+ signatures. Cryptology ePrint Archive (2025)
20. Chase, M., Lysyanskaya, A.: On signatures of knowledge. In: Advances in Cryptology-CRYPTO 2006: 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006. Proceedings 26. pp. 78–96. Springer (2006)
21. Chaum, D.: Security without identification: Transaction systems to make big brother obsolete. Communications of the ACM **28**(10), 1030–1044 (1985)
22. Crites, E.C., Lysyanskaya, A.: Delegatable anonymous credentials from mercurial signatures. In: Cryptographers' Track at the RSA Conference. pp. 535–555. Springer (2019)
23. Desmoulins, N., Dumanois, A., Kane, S., Traoré, J.: Making bbs anonymous credentials eidas 2.0 compliant. Cryptology ePrint Archive (2025)
24. Doerner, J., Kondi, Y., Lee, E., Shelat, A., Tyner, L.: Threshold bbs+ signatures for distributed anonymous credential issuance. In: 2023 IEEE Symposium on Security and Privacy (SP). pp. 773–789. IEEE (2023)
25. Consolidated text: Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 (General Data Protection Regulation) (Text with EEA relevance). `http://data.europa.eu/eli/reg/2016/679/2016-05-04` (2016), `http://data.europa.eu/eli/reg/2016/679/2016-05-04`
26. Amendments by the European Parliament to the Commission 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. `https://www.europarl.europa.eu/doceo/document/A-9-2023-0038_EN.html` (03 2023), `https://www.europarl.europa.eu/doceo/document/A-9-2023-0038_EN.html`
27. Fett, D., Yasuda, K., Campbell, B.: Selective Disclosure for JWTs (SD-JWT). Internet-Draft draft-ietf-oauth-selective-disclosure-jwt-12, Internet Engineering Task Force (Sep 2024), `https://datatracker.ietf.org/doc/draft-ietf-oauth-selective-disclosure-jwt/12/`, work in Progress
28. Flamini, A., Gangemi, A., Guglielmino, E., Orabona, V.: On Delegation of Verifiable Presentations. In: Proceedings of the 3rd International Workshop on Trends in Digital Identity (TDI 2025). pp. 26–38 (2025), `http://ceur-ws.org/Vol-3968/paper2.pdf`
29. Flamini, A., Lee, E., Lysyanskaya, A.: Multi-holder anonymous credentials from bbs signatures. In: Annual International Cryptology Conference. pp. 325–357. Springer (2025)
30. Flamini, A., Sciarretta, G., Scuro, M., Sharif, A., Tomasi, A., Ranise, S.: On cryptographic mechanisms for the selective disclosure of verifiable credentials. Journal of Information Security and Applications **83**, 103789 (2024)
31. Friedrichs, K., Harding, F., Lehmann, A., Lysyanskaya, A.: Device-bound anonymous credentials with (out) trusted hardware. Cryptology ePrint Archive (2025)
32. Frigo, M., et al.: Anonymous credentials from ecdsa. Cryptology ePrint Archive (2024)
33. Griffy, S., Lysyanskaya, A., Mir, O., Perez Kempner, O., Slamanig, D.: Delegatable anonymous credentials from mercurial signatures with stronger privacy. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 296–325. Springer (2024)
34. Hesse, J., Singh, N., Sorniotti, A.: How to bind anonymous credentials to humans. In: 32nd USENIX Security Symposium (USENIX Security 23). pp. 3047–3064 (2023)
35. ISO/IEC 18013-5 Personal identification - ISO-compliant driving licence - part 5: Mobile driving licence (mDL) application (09 2021), `https://www.iso.org/standard/69084.html`
36. Katz, J., Lindell, Y.: Introduction to Modern Cryptography, Second Edition. Chapman & Hall/CRC, 2nd edn. (2014)
37. Katz, J., Raykova, M., Schlesinger, S.: Anonymous credentials with range proofs and rate limiting (2025)
38. Lehmann, A., Sidorenko, A., Zacharakis, A.: Vision: A modular framework for anonymous credential systems. Cryptology ePrint Archive (2025)
39. Mayrhofer, R., Lehmann, A., et al.: A brief note on cryptographic pseudonyms for anonymous credentials. arXiv preprint arXiv:2510.05419 (2025)

40. Orrù, M.: Revisiting keyed-verification anonymous credentials. Cryptology ePrint Archive (2024)
41. Pointcheval, D., Sanders, O.: Short randomizable signatures. In: Topics in Cryptology-CT-RSA 2016: The Cryptographers' Track at the RSA Conference 2016, San Francisco, CA, USA, February 29-March 4, 2016, Proceedings. pp. 111–126. Springer (2016)
42. Sporny, M., Longley, D., Chadwick, D.: Verifiable Credentials Data Model (03 2022), `https://www.w3.org/TR/vc-data-model/`
43. Tessaro, S., Zhu, C.: Revisiting BBS signatures. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 691–721. Springer (2023)

Appendix

.1 Unforgeability and Strong Unforgeability

Unforgeability. The notion of *unforgeability* for digital signature schemes is defined via the following experiment between a challenger and an adversary $\mathcal{A}$:

Setup: The challenger runs the key generation algorithm KeyGen to obtain a public key pk and a private key sk. The adversary is given pk.
Queries: The adversary requests signatures on at most q_S messages of its choice $M_1, \ldots, M_{q_S} \in \{0,1\}^*$ under pk. The challenger responds to each query with a signature $\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}, M_i)$.
Output: The adversary outputs a pair (M, σ) and wins the game if:

1. M is not equal to any of $M_1, \ldots, M_{q_S}$;
2. $\mathsf{Vf}(\sigma, M, \mathsf{pk}) = 1$.

We denote by $\mathsf{SigAdv}_{\mathcal{A}}$ the probability that adversary $\mathcal{A}$ wins the above experiment, over the randomness of both $\mathcal{A}$ and the challenger.

Definition 18 (Unforgeable signature scheme). *A signature scheme* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vf})$ *is said to be* unforgeable *if, for all probabilistic polynomial-time adversaries $\mathcal{A}$, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$\mathsf{SigAdv}_{\mathcal{A}} \leq \mathsf{negl}(\lambda),$$

where λ is the security parameter.

Strong Unforgeability. The notion of *strong unforgeability* for digital signature schemes [10] is defined via the following experiment between a challenger and an adversary $\mathcal{A}$:

Setup: The challenger runs the key generation algorithm KeyGen to obtain a public key pk and a private key sk. The adversary is given pk.
Queries: The adversary requests signatures on at most q_S messages of its choice $M_1, \ldots, M_{q_S} \in \{0,1\}^*$ under pk. The challenger responds to each query with a signature $\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}, M_i)$.
Output: The adversary outputs a pair (M, σ) and wins the experiment if:

1. (M, σ) is not equal to any of $(M_1, \sigma_1), \ldots, (M_{q_S}, \sigma_{q_S})$;
2. $\mathsf{Vf}(\sigma, M, \mathsf{pk}) = 1$.

We denote by $\mathsf{StrongSigAdv}_{\mathcal{A}}$ the probability that adversary $\mathcal{A}$ wins the above experiment, over the randomness of both $\mathcal{A}$ and the challenger.

Definition 19 (Strongly unforgeable signature scheme).
A signature scheme $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vf})$ *is said to be* strongly unforgeable *if, for all probabilistic polynomial-time adversaries $\mathcal{A}$, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$\mathsf{StrongSigAdv}_{\mathcal{A}} \leq \mathsf{negl}(\lambda),$$

where λ is the security parameter.

.2 Anonymous Credential Scheme

An *anonymous credential scheme* is defined by a set of algorithms and protocols that establish secure processes to issue, present and verify the AC.

Definition 20. *Let $\mathcal{DS} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vf})$ be a digital signature scheme and H a cryptographic hash function. An* anonymous credential scheme *is defined by the following algorithms:*

- Issuer setup: $\{\mathsf{pp}, (\mathsf{sk}_{\mathsf{Iss}}, \mathsf{pk}_{\mathsf{Iss}})\} \overset{\$}{\leftarrow} \mathsf{IssuerSetup}(\lambda)$.
  The issuer executes the IssuerSetup *algorithm which consists in executing the* Setup *and* KeyGen *algorithms of the underlying digital signature scheme $\mathcal{DS}$. This algorithm generates:*
  - *the public parameters:* $\mathsf{pp} \overset{\$}{\leftarrow} \mathsf{Setup}(\lambda)$;
  - *the issuer key pair:* $(\mathsf{sk}_{\mathsf{Iss}}, \mathsf{pk}_{\mathsf{Iss}}) \overset{\$}{\leftarrow} \mathsf{KeyGen}(\mathsf{pp})$.
- Credential issuance: $\mathsf{cred} \overset{\$}{\leftarrow} \mathsf{CredIssuance}(\{a_i\}_{i\in[l]}, \mathsf{sk}_{\mathsf{Iss}})$.
  The issuer executes the following operations:
  - *sign the attributes $\{a_i\}_{i\in[l]}$ using their private key $\mathsf{sk}_{\mathsf{Iss}}$, computing $\sigma \overset{\$}{\leftarrow} \mathsf{Sign}(\{a_i\}_{i\in[l]}, \mathsf{sk}_{\mathsf{Iss}})$;*
  - *set* $\mathsf{cred} \leftarrow (\sigma, \{a_i\}_{i\in[l]})$.
- Credential presentation: $\pi \overset{\$}{\leftarrow} \mathsf{CredPresentation}(\mathsf{cred}, \mathsf{stmt}, \mathsf{nonce})$.
  The statement $\mathsf{stmt} = \{a_i\}_{i\in\mathsf{Rev}}$ is given by the set of attributes that the holder wants to reveal $\mathsf{Rev} \subseteq [l]$, and the nonce *is sent by the verifier to the holder to guarantee the freshness of the presentation. The holder constructs a non-interactive zero-knowledge proof π that demonstrates that the attributes $\{a_i\}_{i\in\mathsf{Rev}}$ are included in* anoncred*. To do that, the holder:*
  - *given $(\sigma, \{a_i\}_{i\in[l]})$, generates a commitment t;*
  - *computes the challenge $c \leftarrow H(t, \mathsf{nonce})$;*
  - *computes a response z, and set $\pi = (t, z)$.*
- Presentation Verification: $\{0, 1\} \overset{\$}{\leftarrow} \mathsf{PresVer}(\pi, \mathsf{stmt})$.
  The verifier performs the following operations:
  - *computes $c = H(t, \mathsf{nonce})$;*
  - *checks whether (t, c, z) is an accepting transcript for $\{a_i\}_{i\in\mathsf{Rev}}$.*

  If the previous checks are satisfied, the verifier accepts and outputs 1, otherwise it outputs 0.

.3 *q*-Strong Diffie–Hellman Assumption

In Theorem 4, proving the unforgeability of the delegation scheme described in Definition 16, we rely on the strong unforgeability of BBS signatures, which holds under the q-SDH assumption in bilinear groups. For completeness, we recall the definition of the q-Strong Diffie–Hellman (q-SDH) assumption [5].

Definition 21 (q-Strong Diffie–Hellman (q-SDH) Assumption). *Let $(\mathbb{G}_1, \mathbb{G}_2)$ be cyclic groups of prime order p with generators $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$, and let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be an efficiently computable bilinear pairing. The q-*Strong Diffie–Hellman (q-SDH) problem *in $(\mathbb{G}_1, \mathbb{G}_2)$ is defined as follows: Given the $(q + 2)$-tuple*

$$(g_1, g_2, g_2^x, g_2^{x^2}, \ldots, g_2^{x^q}) \in \mathbb{G}_1 \times \mathbb{G}_2^{q+1}$$

for some randomly chosen secret $x \in \mathbb{Z}_p^*$, compute a pair (A, c) such that

$$A^{x+c} = g_1 \quad \text{where } A \in \mathbb{G}_1,\ c \in \mathbb{Z}_p^*.$$

We say that the (q, t, ε)-SDH assumption holds in $(\mathbb{G}_1, \mathbb{G}_2)$ if no algorithm running in time t has advantage at least ε in solving the q-SDH problem in $(\mathbb{G}_1, \mathbb{G}_2)$.

.4 Collision Handling in the Random Oracle Simulation

In the proof of Theorem 5, we constructed a simulator $\mathcal{S}$ that emulates the delegated presentation by programming the random oracle RO. The correctness of this simulation relies on the ability of $\mathcal{S}$ to consistently answer all random oracle queries without overwriting previously assigned outputs in the internal hash table HT. The purpose of this section is to rigorously analyze this event and to show that the probability of such a collision occurring is negligible in the security parameter λ.

Probability of Collision The adversary $\mathcal{A}$ is allowed to make a polynomial number of queries to RO. For each new random oracle query for input x made by $\mathcal{A}$, the simulator $\mathcal{S}$ samples a random digest $c \xleftarrow{\$} \mathbb{Z}_p$ and stores the pair (x, c) in a hash table HT. Also, during the simulation of a delegation presentation, the simulator $\mathcal{S}$ generates a random digests that will be the challenges $c \in \mathbb{Z}_p$ used to generate the simulated proof and gives it as an input to the HVZK simulator of the sigma protocol for the presentation to compute the first message U. Once U is computed, $\mathcal{S}$ programs the random oracle setting $\mathsf{HT}(\bar{A}, \bar{B}, U, \mathsf{del}, \mathsf{nonce}) = c$. If $\mathcal{A}$ has previously queried RO with this exact input, and the random oracle has already assigned a different value $c' \neq c$ to this input, then the simulator $\mathcal{S}$ fails in simulating the unforgeability experiment the hash table and the simulation fails.

Let $Q_1, \ldots, Q_q$ denote the q distinct queries made by $\mathcal{A}$ to RO (with q polynomial in λ). The simulator fails if, for some $j \in [q]$, the an input $Q_j = (\bar{A}_j, \bar{B}_j, U_j, \mathsf{del}_j, \mathsf{nonce}_j)$ matches the tuple $(\bar{A}', \bar{B}', U', \mathsf{del}', \mathsf{nonce}')$ constructed by the simulator in the simulation of the presentation.

Since del and nonce are chosen by the adversary, while $\bar{A}', \bar{B}', U'$ are generated by the simulator, the probability that the simulator generates a tuple that matches one of the adversary's previous queries can be bounded computing the probability that the adversary guesses $\bar{A}', \bar{B}', U'$.

In the simulation $\bar{A}'$ is distributed uniformly in $\mathbb{G}_1$, $\bar{B}'$ is uniquely determined by $\bar{A}'$, U' is again distributed uniformly at random in $\mathbb{G}_1$, $|\mathbb{G}_1| = p$,

$$\Pr[\text{Simulator fails in a presentation query}] \leq \sum_{j=1}^{q} \Pr[(\bar{A}, \bar{B}, U, \mathsf{del}, \mathsf{nonce}) = Q_j] \leq \frac{q}{p^2}.$$

Since $q = \text{poly}(\lambda)$ and p is exponential in the security parameter λ, the probability that the Simulator fails is negligible in λ.