# **Secure Showing of Partial Attributes**

Foteini Baldimtsi George Mason University and Mysten Labs Fairfax, VA, USA foteini@gmu.edu

Julian Loss CISPA Helmholtz Center for Information Security Saarbrucken, Germany loss@cispa.de

# ABSTRACT

Anonymous Attribute-Based Credentials (ABCs) allow users to prove possession of attributes while adhering to various authentication policies and without revealing unnecessary information. Single-use ABCs are particularly appealing for their lightweight nature and practical efficiency. These credentials are typically built using blind signatures, with Anonymous Credentials Light (ACL) being one of the most prominent schemes in the literature. However, the security properties of single-use ABCs, especially their secure showing property, have not been fully explored, and prior definitions and corresponding security proofs fail to address scenarios involving partial attribute disclosure effectively. In this work, we propose a stronger secure showing definition that ensures robust security even under selective attribute revelation. Our definition extends the winning condition of the existing secure showing experiment by adding various constraints on the subsets of opened attributes. We show how to represent this winning condition as a matching problem in a suitable bipartite graph, thus allowing for it to be verified efficiently. We then prove that ACL satisfies our strong secure showing notion without any modification. Finally, we define double-spending prevention for single-use ABCs, and show how ACL satisfies the definition.

## **KEYWORDS**

Blind Signatures, Anonymous Credentials

## **1** INTRODUCTION

Anonymous Credentials  $(ACs)^1$ , are a fundamental cryptographic primitive that enable a user U to authenticate anonymously to a verifier V. In AC systems, when U subscribes to some service, an issuer S blindly issues a token to U as proof of subscription. Subsequently, U can redeem this token without linking itself to a particular prior session with S (even if S and V are the same party or collude with each other) in a way that might give away additional unwanted metadata about U. Beyond enabling basic privacy-preserving authentication, ACs can be augmented to include additional user data for U such as age, location, and other attributes, collectively referred to as *attributes*. Anonymous Attribute-Based Credentials (ABCs) extend ACs by incorporating a more sophisticated showing procedure, which allows users to selectively present subsets of these attributes along with the credential (*partial reveal*). This makes it possible to Julia Kastner Centrum Wiskunde & Informatica Amsterdam, Netherlands julia.kastner@cwi.nl

Omar Renawi CISPA Helmholtz Center for Information Security Saarland University Saarbrucken, Germany omar.renawi@cispa.de

implement fine-grained authentication policies without compromising *U*'s privacy. Numerous ABC protocols have been proposed in the literature [3, 5, 7–10, 13, 16, 17, 19, 22, 24, 31, 41, 44] and related systems have been adopted practice (i.e. in Signal [20], or in the context of Direct <u>Anonymous Attestation (DAA) and Enhanced</u> <u>Privacy ID (EPID)).</u>

While some ABCs allow for a high degree of reusability of an issued token across multiple authentications, they often rely on expensive pairing operations or computationally intensive zeroknowledge machinery. In contrast, *single-use* ABCs provide a lightweight alternative that ensures anonymity for a single use while still supporting important applications. A simplified variant of singleuse ABCs, which does not support attributes, is referred to as anonymous tokens. These tokens have gathered growing interest both in the academic literature [6, 18, 25, 38, 40, 45] but also among major companies such as Cloudfare,<sup>2</sup> Apple,<sup>3</sup>, Google,<sup>4</sup> and Facebook<sup>5</sup>. They are also being standardized in an IETF draft<sup>6</sup>.

Despite the growing interest in single-use token protocols, proposed schemes are designed as basic, one-time-use tokens that prioritize simplicity and efficiency but do not incorporate user attributes into the issued tokens. This limitation is significant, as the ability to include attributes-and to selectively reveal only the necessary ones-is crucial for applications where users require precise control over the specific data they wish to disclose. One notable exception is the construction by Baldimtsi and Lysyanskaya of Anonymous Credentials Light (ACL) [3], which effectively serves as a single-use token with attributes. However, as we detail below, the security of ACL remains poorly understood, particularly in scenarios where users need to selectively reveal attributes embedded in the token. This gap highlights an interesting research question: do efficient, and provably secure single-use tokens with attributes and support for partial attribute reveal exist? We answer this question in the affirmative by introducing a rigorous security definition for single-use tokens with attributes and partial reveal, and by proving that the existing ACL scheme satisfies this definition.

<sup>&</sup>lt;sup>1</sup>Also referred to as anonymous tokens in some cases.

<sup>&</sup>lt;sup>2</sup>https://blog.cloudflare.com/privacy-pass-standard

<sup>&</sup>lt;sup>3</sup>https://developer.apple.com/news/?id=huqjyh7k

<sup>&</sup>lt;sup>4</sup>https://github.com/google/anonymous-tokens,

https://developers.google.com/privacy-sandbox/protections/private-state-tokens <sup>5</sup>https://research.fb.com/privatestats

<sup>&</sup>lt;sup>6</sup>https://datatracker.ietf.org/wg/privacypass/about/

Blind Signatures with Attributes and the ACL Scheme. Anonymous Credentials Light (ACL)[3] is one of the most popular singleuse ABC systems. ACL provides a single-use anonymous credential system with attributes from basic group-based cryptography (in particular without the use of pairings). Although the ACL construction was first published over a decade ago, its delicate security properties have only recently begun to be understood. In their original work, Baldimtsi and Lysyanskaya presented a useful abstraction of ABCs called Blind Signatures with Attributes (BSAs) which captures the essential properties of single-use ABCs with minimal syntactical overhead. In a BSA, S is abstracted as a signer S which holds a secret key sk for which the user *U* holds a corresponding public key pk. During a registration phase, S checks U's attributes, and U commits to them using a commitment C. In the subsequent singing phase, S blindly issues to U a credential in the form of a signature  $\sigma$  on a rerandomization of C, which can be verified against pk at the time of showing. Baldimtsi and Lysyanskaya also introduced an appropriate security notion to capture the properties of BSAs. They defined it through an experiment which on one hand, encompasses the standard one-more unforgeability property (which states that U should not be able to obtain a valid signature without interacting with S) and on the other hand, it also ensures that U cannot alter the attributes that it committed to during the registration phase.

Revisiting the Security Properties of ACL. Unfortunately, the original security proof of ACL falls short of achieving this security property in two aspects. First, the proof guarantees security only as long as S issues signatures in a sequential fashion. Second, Baldimtsi and Lysyanskaya do not formally prove the security properties related to the secure showing aspects of their definition. In very recent work, Kastner et al. [35] resolve both of these issues by giving the first proof of security (unforgeability) for ACL under concurrent issuing sessions and formally proving its secure showing property. In the process, they also provide a new, modular security definition which separates the security of ACL's showing procedure from its unforgeability. In doing so, however, their work also uncovers a subtlety with the original showing security definition of Baldimtsi and Lysyanskaya. In their security experiment, the adversary interacts with the signer and obtains a set of  $\ell$  signatures  $\sigma_1, \ldots, \sigma_\ell$ with corresponding attribute vectors  $\vec{L}'_1, \ldots, \vec{L}'_{\ell}$ . The adversary is considered successful in breaking the secure showing property if, at the end of the experiment, it can produce an alternative multiset of vectors  $\vec{L}_1, \ldots, \vec{L}_\ell$  which differs from the original one in at least one component, yet still verifies against  $\sigma_1, \ldots, \sigma_\ell$ . (We consider multisets as the same attribute vector may appear multiple times among  $\vec{L}'_1, \ldots, \vec{L}'_{e}$ .) While this intuitively ensures that the adversary cannot alter its attributes post-facto, closer scrutiny reveals that this definition does not cover the natural scenario in which the user only reveals a subset of its attributes in the showing procedure (partial reveal). This option of partial attribute showing is indeed provided by the syntax of the showing procedure. However, the security experiment corresponding to the showing definition only allows for the full set of attributes to be shown.

A Strong Secure Showing Definition. As the main contribution of this work, we formalize a stronger secure showing definition for blind signatures with attributes which remedies the aforementioned shortcomings by supporting partial showing of attributes and prove that ACL satisfies it without modification. As we will now discuss, this presents several unexpected subleties both at a definitional and technical level.

In the new notion, the adversary can again query the challenger of the security experiment for signatures corresponding to attribute vectors  $\vec{L}'_1, \ldots, \vec{L}'_k$ . Similar to the original definition, the adversary has to provide, at the end of the experiment, a set of vectors  $\vec{L}_1, \ldots, \vec{L}_\ell$  which each verify against at least one of the signatures  $\sigma_i$ . However, different from before,  $\vec{L}_i$  may now be *partial* vectors, that is some entries can be replaced by a special empty symbol  $\perp$  (for honest users, these partial vectors would be subvectors of the previously signed vectors  $\vec{L}'_i$ , i.e. they would agree on the non- $\perp$  entries).

The winning condition of our modified experiment intuitively has to check whether the set of opened partial vectors cannot be explained by a set of honest users opening their signatures. For such an honest set of users, there would be an explanation for each of the opened vectors  $\vec{L}_i$  by a signing session with a vector  $\vec{L}'_i$  such that  $\vec{L}_i$  is a subvector of  $\vec{L}'_i$  and no signing session is used to explain more than one opened signature. More formally, the game checks if there is an *injective* mapping from the opened (partial) vectors  $\vec{L}_1, \ldots, \vec{L}_\ell$  to the signed vectors  $\vec{L}'_1, \ldots, \vec{L}'_k$  (multiply signed vectors appear multiple times) such that the opened vectors are subvectors of the corresponding signed vectors. If no such mapping exists, the adversary has won the game, as it must have altered at least one of the signed vectors post-facto. Alternatively, this means for any attempt at explaining the opened partial vectors through the signatures issued, there is at least one signature with opened vector  $\vec{L}_i$  that cannot be explained by a signing session.

For example, consider an adversary that queried the challenger for one signature corresponding to  $\vec{L}'_1 = (a, b)$  and two signatures corresponding to  $\vec{L}'_2 = \vec{L}'_3 = (a, c)$ . If it outputs three signatures  $\sigma_1, \sigma_2, \sigma_3$  with (verifying) corresponding partial openings to attributes  $(a, c), (\bot, c), (\bot, c)$ , it wins this game. Namely, since both (a, c) and  $(\cdot, c)$  are subvectors only to  $\vec{L}'_2 = \vec{L}'_3$ , but not  $\vec{L}'_1$ , it is impossible to assign the three openings as subvectors injectively to the (multi-)set of previously signed attribute vectors  $\{\vec{L}'_1, \vec{L}'_2, \vec{L}'_3\}$ . On the other hand, an adversary that opens the signatures to  $(a, \bot), (\bot, c), (\bot, c)$  would not win the game, as the  $(a, \bot)$  now constitutes a subvector to any of  $\vec{L}'_1, \vec{L}_2$  and  $\vec{L}'_3$ , and the opened vectors can be explained by  $(a, \bot) \mapsto \vec{L}'_1, (\bot, c) \mapsto \vec{L}'_2, (\bot, c) \mapsto \vec{L}'_3$ . Unfortunately, this new notion introduces an entirely different

Unfortunately, this new notion introduces an entirely different issue: it is unclear whether this new secure showing definition is satisfiable *at all*, as it seemingly asks the challenger to check an exponentially complex set of constraints related to the winning condition of the security experiment.

**Strong Secure Showing for ACL.** To overcome this technical difficulty, we propose the following generic blueprint. First, one abstracts the secure showing experiment as a bipartite graph *G* where the sets of nodes correspond to the attribute vectors  $\vec{L}_1, \ldots, \vec{L}_\ell$  and  $\vec{L}'_1, \ldots, \vec{L}'_\ell$  and an edge between  $\vec{L}_i$  and  $\vec{L}'_j$  indicates that  $\vec{L}'_j$  is a subvector of  $\vec{L}_i$ . We observe that we can view the winning condition as the non-existence of a matching between the node sets in *G*.

Thus, in the second step one can leverage the Hopcroft-Karp algorithm [33] to efficiently find such a maximal matching in *G*. If the algorithm comes up short, i.e., leaves one of the nodes corresponding to  $\vec{L}_1, \ldots, \vec{L}_\ell$  unmatched, the adversary wins the game. Unfortunately, it is unclear how to proceed from here for the general case. In more detail, the simple fact that the we can *detect* when an adversary wins the the security experiment does not mean that we can turn such an adversary into an efficient routine for breaking the hardness assumption underlying the schemes' security.

Thus, for ACL, we take a slightly different route. An ACL-Sign ature contains a component that is a "blinded" version of the registered commitment. We first argue in the Algebraic Group Model (AGM), using similar techniques as [35], that there exists an injective mapping from message-signature pairs to signing sessions. We then use this mapping as the base for our matching of partial opening vectors to full registered vectors. For this, we provide an extraction algorithm for the blinding factors of the commitments in the AGM. If the adversary wins the game, there must be a vector that cannot be matched using matching algorithms, and thus there must be at least one message-signature pair where the partially opened vector is not a subvector of the corresponding full registered vector used in the signing session. We use further AGM-based extraction to extract a second opening of the registered commitment, thus breaking the binding property of the commitment scheme.

Add-On: Double-Spending Detection When using single-use ABCs one should be particularly careful on how to enforce the single-use property. The simplest solution is to require the verification phase to be "online", i.e., assuming multiple verification services, there exists a centralized authority that tracks previously used/redeemed tokens and during verification the receiver checks with the authority whether the token is new. However, this is often not possible in constrained environments which might require fast verification while suffering from poor connectivity (e.g. in the setting of public transportation payments). In such scenarios, a double-spending detection mechanism is preferred where verification can happen offline but later it can be checked whether the same token was used and identify the misbehaving user.

We extend the blind signature with attributes definition to also support properties for double-spending protection enabling it to more naturally address all the requirements of single-use anonymous credentials or tokens. Specifically, we first define a security property that ensures a misbehaving user who presents the same token more than once can be identified. Second, we define an exculpability property that guarantees a malicious signer cannot falsely accuse an honest user of double spending.

Then, we provide a simple modification of the ACL scheme to satisfy our definitions. The modification is based on a similar protocol proposed by Abe [1] for the blind signature scheme ACL is based on. This same scheme was also implemented for ACL in [32] but without defining or proving its security. The key idea is to add the first flow of a sigma-protocol to be used in signature showing to the signature in such a way that it cannot be removed/exchanged, namely by appending it in the string that is hashed at signature generation time. Then, the showing protocol also involves proving knowledge of the user's attributes using this first flow contained in the signature. We implement this using the Fiat-Shamir (FS) heuristic. We note that both the user and the signer include a current timestamp to derive their FS challenge. This is important in order to ensure that if the user shows the same signature twice with different timestamps, the challenge from the hash function will be different. As the user has to always use the same first flow contained in the signature, the two different transcripts reveal his identity.

#### 1.1 Related Work

Anonymous Credentials (ACs) were originally proposed by David Chaum [22] and later significantly improved by Camenisch and Lysyanskaya [16, 17]. Since then, they have been the focus of extensive research, have seen adoption in industrial applications, and align well with governmental policies emphasizing privacypreserving digital identity frameworks and data minimization principles.

As noted, a variety of constructions have been proposed in the literature, each secure under different cryptographic assumptions and designed to address a distinct set of properties tailored to specific use cases. Two important axes of distinction are whether they support multi-use or single-use showing and whether they include support for attributes.

Typically, multi-use credentials support attributes due to their more permanent nature and richer set of capabilities. A variety of such protocols exist in the literature, the most popular being based on discrete log and bilinear pairings and/or variations of the RSA problem [7–9, 13, 16, 17, 31, 44] and more recently on lattice assumptions [10]. There have also been proposals on extensions that support credential delegation [5, 19, 24], decentralized issuance [26, 30, 46] and revocation [2, 12, 15, 16].

On the single-use tokens side, most of the proposals do not support user attributes [6, 18, 25, 38, 40, 45]. In this category, can we also include anonymous e-cash schemes [11, 14, 21, 29, 39, 43] (where one can treat every digital coin as a single-use token). Extensions on decentralized issuance were also proposed [34, 47]. Existing works on single-use credentials with attributes that do support attributes [3, 11, 41] lack proper security proofs<sup>7</sup>.

#### 2 PRELIMINARIES AND DEFINITIONS

Notation. For a natural number n, we denote by [n] the set of numbers  $\{1, \ldots, n\}$  and by  $[n]_0$  the set  $\{0\} \cup [n]$ . Throughout the paper, we fix a group  $\mathbb{G}$  of prime order p with generator g and assume that all algorithms have access to this group's parameters even when not given as an explicit input. We use := for deterministic assignments and  $\leftarrow_{\$}$  to denote that a value is sampled at random from a set or the output of a randomized algorithm. We denote vectors  $\vec{L}$  and by  $L_i$  the  $i_{th}$  entry of  $\vec{L}$ . We say a vector  $\vec{L}'$  is a *partial* vector if for some indices  $i, L'_i = \bot$  where  $\bot$  is a special empty symbol. For a vector  $\vec{L}$  of length n and a (partial) vector  $\vec{L}'$ , we say  $\vec{L}'$  is a *subvector* of  $\vec{L}$ , denoted by  $\vec{L}' \subseteq L$ , if for each index  $i \in [n]$  it holds that either  $L'_i = L_i$  or  $L'_i = \bot$ . A multiset is a generalization of a set where each element can appear multiple times. We call the number of times an element x appears in a multiset M the *multiplicity* of x. For multisets M, M' we say that M' is a *subset* of

<sup>&</sup>lt;sup>7</sup>The case of why ACL lacks the required formal proofs was extensively discussed above. The issues with the security of the schemes based on Brands'[11, 41] were previously discussed in [4].

M, denoted by  $M' \subseteq M$ , if each element in M' is also contained in M where the multiplicity of each element contained in both sets is lower or equal for M' than for M.

In the following sections, we assume that there exist public parameters pp that are known to all algorithms even when not explicitly given as an input.

# 2.1 Cryptographic Background

Definition 2.1 (AGM/algebraic algorithms [28]). We say that an algorithm is algebraic with respect to a group  $\mathbb{G}$  if it takes as input a set of group elements  $g_1, \ldots, g_n$  (w.l.o.g. we assume that one of them is the generator g and in case of interactive algorithms group elements in the output of any potential oracles are added to the set) along with a bit string input s and for every group element y that it outputs it also outputs an explanation vector (sometimes also called the representation)  $\vec{z}$  such that

$$y = \prod_{i=1}^{n} g_i^{z_i}.$$

In the *algebraic group model (AGM)* it is assumed that every algorithm is algebraic.

*Definition 2.2 (Discrete Logarithm Problem (DLP)).* We define the discrete logarithm game  $DLP_{\mathbb{G}}$  for the group  $\mathbb{G}$  with generator g and prime order p as follows:

**Setup.** Sample  $x \leftarrow_{\$} \mathbb{Z}_p$ . Invoke the adversary on input  $g, g^x$ . **Output Determination.** The adversary outputs a candidate solution  $x' \in \mathbb{Z}_p$ , the game outputs 1 if  $g^{x'} = y, 0$  otherwise.

We define the advantage of an adversary  $\mathcal R$  in the game DLP as

$$\operatorname{Adv}_{\mathbb{G}}^{\operatorname{DLP}}(\mathcal{A}) \coloneqq \Pr[\operatorname{DLP}_{\mathbb{G}}^{\mathcal{A}} = 1]$$

and say that DLP is  $(t, \varepsilon)$ -hard in the group  $\mathbb{G}$  if for any adversary  $\mathcal{A}$  that runs in time at most t, it holds that  $\operatorname{Adv}_{\mathbb{G}}^{\operatorname{DLP}}(\mathcal{A}) \leq \varepsilon$ 

#### 2.2 Commitment Schemes

Definition 2.3 (Commitment Scheme). A commitment scheme  $CS = (\mathcal{PG}, C, \mathcal{V})$  consists of the following algorithms:

- The *parameter generation algorithm*  $\mathcal{PG}$  takes as input public parameters pp and outputs commitment parameters pp<sub>CS</sub>.
- The *commit algorithm C* takes as input commitment parameters pp<sub>CS</sub> and a value *v* outputs a commitment *C* and an opening *r*.
- The *verification algorithm CV* takes as input commitment parameters pp<sub>CS</sub>, a commitment *C*, a value *v* and an opening *r* outputs either 1 (ACCEPT) or 0 (REJECT).

*Definition 2.4 (Hiding).* We define the HIDING game for a commitment scheme CS and an adversary  $\mathcal{A}$  as follows:

- **Setup:** Run CS. $\mathcal{PG}(pp)$  to obtain  $pp_{CS}$ , output CS to the adversary. Sample a bit *b* uniformly at random.
- **Online Phase:** The adversary outputs two values  $v_0, v_1$ . The game computes  $(C, r) \leftarrow_{\$} CS.C(pp_{CS}, v_b)$  and outputs *C* to the adversary.
- **Output Determination:** The adversary outputs a bit b'.

We define the advantage of the adversary  $\mathcal R$  as

$$\operatorname{Adv}_{\operatorname{CS}}^{\operatorname{HIDING}}(\mathcal{A}) \coloneqq 2 \cdot \left| \Pr[b = b'] - \frac{1}{2} \right|$$

We say that CS is  $(\varepsilon, t)$ -*hiding* if for any adversary  $\mathcal{A}$  that runs in time at most *t*, it holds that  $Adv_{CS}^{HIDING}(\mathcal{A}) \leq \varepsilon$ . We say that CS is *perfectly hiding* if for any adversary the advantage is 0.

*Definition 2.5 (Binding).* We define the **BINDING** game for a commitment scheme CS and an adversary  $\mathcal{A}$  as follows:

- **Setup:** Run CS. $\mathcal{PG}$  to obtain pp<sub>CS</sub>, output CS to the adversary.
- **Output Determination:** The adversary outputs a commitment *C*, two values  $v_1, v_2$  and two openings  $r_1, r_2$ . The game outputs 1 if  $v_1 \neq v_2$  and  $CV(pp_{CS}, v_1, r_1) = CV(pp_{CS}, v_2, r_2) = 1$ .

We define the advantage of the adversary  ${\mathcal A}$  as

$$\mathbf{Adv}_{\mathsf{CS}}^{\mathsf{BINDING}}(\mathcal{A}) \coloneqq \Pr\left[\mathbf{BINDING}_{\mathsf{CS}}^{\mathcal{A}} = 1\right]$$

and we say that CS is  $(t, \varepsilon)$ -binding if for any adversary  $\mathcal{A}$  that runs in time at most t it holds that  $\operatorname{Adv}_{CS}^{\operatorname{BINDING}}(\mathcal{A}) \leq \varepsilon$ . We say that CS is *perfectly binding* if for any adversary the advantage is 0.

Definition 2.6 (Generalized Pedersen Commitment [42]). We describe the generalized Pedersen Commitment Scheme GPC over a group  $\mathbb{G}$  described by group parameters  $pp_{\mathbb{G}}$  (which take the role of the public parameters pp) for vectors of length *n* below:

 $\mathcal{PG}(pp_{\mathbb{G}})$ : sample n + 1 group elements  $h_0, \ldots, h_n \leftarrow_{\$} \mathbb{G}$ . Output those group elements as the commitment parameters  $pp_{\text{GPC}} = (h_0, \ldots, h_n)$ .

 $C(\text{pp}_{\text{GPC}}, \vec{v})$ : Given commitment parameters as above and a vector  $\vec{v} \in \mathbb{Z}_p^n$  with *n* entries, sample an exponent  $r \leftarrow_{\$} \mathbb{Z}_p$  and output  $C = h_0^r \prod_{i=1}^n h_i^{v_i}$  and *r*.

 $C\mathcal{V}(pp_{GPC}, v, C, r)$ : Given the commitment parameters as above, a value v, a commitment C, and an opening r, output 1 iff  $C = h_0^r \prod_{i=1}^n h_i^{v_i}$ .

Definition 2.7 (PoK for Opening of a Generalized Pedersen Commitment). We describe a protocol for proving knowledge of a full opening of GPC as described in definition 2.6, i.e. for the relation  $\mathcal{R}_{\vec{h}} = \{(C, (r, \vec{v})) : h_0^r \cdot \prod_{i=1}^n h_i^{v_i} = C\},\$ 

- The prover algorithm  $\mathcal{P}$  is split into two algorithms  $\mathcal{P}_1$  and  $\mathcal{P}_2$ .
  - $\mathcal{P}_1(\vec{h}, C, (r, \vec{v}))$ : Sample values  $r_0, \ldots, r_n \leftarrow_{\$} \mathbb{Z}_p$ . Compute  $R = \prod_{i=0}^n h_i^{r_i}$ . Output the first flow R and keep the local state  $r_0, \ldots, r_n$ .
  - $\mathcal{P}_2(r_0, \ldots, r_n, c, \vec{v})$ : Compute  $s_0 = r_0 c \cdot r$ , for  $i = 1, \ldots, n$  compute  $s_i = r_i c \cdot v_i$ . Output  $s_0, \ldots, s_n$ .
- The *verifier algorithm* V is split into two algorithms  $V_1$  and  $V_2$ :
  - −  $\mathcal{V}_1(\vec{h}, C, R)$ : Sample a random challenge  $c \leftarrow_{\$} \mathbb{Z}_p$ , output c, keep local state  $\vec{h}, C, R$ .
  - $\mathcal{V}_2(\vec{h}, C, R, c, \vec{s})$ : Output 1 if  $R = C^c \cdot \prod_{i=0}^n h_i^{s_i}$ , 0 otherwise.

We prove the following two lemmas in appendix A.1.

LEMMA 2.8 (PEDERSEN COMMITMENTS ARE PERFECTLY HIDING). The commitment scheme GPC from definition 2.6 is perfectly hiding.

LEMMA 2.9 (PEDERSEN COMMITMENTS ARE BINDING UNDER DL). If the DLP is  $(t, \varepsilon)$ -hard in the group G, GPC from definition 2.6 is  $(t', \varepsilon')$ -binding with  $t' \approx t$  and  $\varepsilon' \leq n \cdot \varepsilon$ .

## 2.3 Blind Signature with Attributes

Blind signatures with attributes were first defined in [3] and the definition was later refined in [35]. Here, we recall the definition as presented in [35].

Definition 2.10. A blind signature with attributes is a tuple of algorithms BSA :=  $(\mathcal{G}, \mathcal{R}, \mathcal{S}, \mathcal{U}, \mathcal{V}, \mathcal{SH})$ , with the following properties.

- The *key generation algorithm G* takes as input public parameters pp and outputs a pair of keys (pk, sk) for the signer. In the rest of the paper we assume that pk implicitly contains a copy of pp and sk implicitly contains a copy of pk.
- The *registration protocol* R between the signer and the user consists of two algorithms R<sub>U</sub> and R<sub>S</sub>:
  - $\mathcal{R}_{\mathcal{U}}$  takes as input a vector of attributes  $\vec{L}$  and outputs a commitment C to  $\vec{L}$ , a registration state  $st_{\mathcal{R}}$ , and a proof  $\pi$ .
  - $\mathcal{R}_{S}$  is deterministic and takes as input a commitment *C*, and a proof  $\pi$ . It outputs 1 (ACCEPT) or 0 (REJECT).
- The *signer algorithm* S is split into two algorithms S<sub>1</sub> and S<sub>2</sub>:
  - $S_1$  takes as input a secret key sk and a commitment *C*. It returns a commitment *R* and a signer state  $st_S$ .
  - S<sub>2</sub> is deterministic and takes as input a secret key sk, a singer state st<sub>S</sub>, a commitment *R*, and a challenge *e*. It returns a response *S*.
- The user algorithm  $\mathcal U$  is split into two algorithms  $\mathcal U_1$  and  $\mathcal U_2$ :
  - $\mathcal{U}_1$  takes as input a public key pk, commitments *R* and *C*, a message *m*, and a registration state  $st_{\mathcal{R}}$ . It returns a challenge *e* and a user state  $st_{\mathcal{U}}$ .
  - $\mathcal{U}_2$  is deterministic and takes as input a public key pk, a user state  $st_{\mathcal{U}}$ , a commitment *R*, a challenge *e*, a response *S*, and a message *m*. It returns a signature  $\hat{\sigma}$  and a user opening state  $st_O$
- The *show algorithm* SH is split into two algorithms SH<sub>U</sub> and SH<sub>V</sub>:
  - The show generation algorithm  $SH_U$  takes as input a public key pk, a registration state  $st_R$  (which includes the attribute vector of the user), a message *m*, signature  $\hat{\sigma}$ , a user's signer session state  $st_O$  and an index set  $I_{op}$ . It outputs a partial vector  $\vec{L}'$  and a proof  $\bar{\pi}$ .
  - The show verification algorithm  $SH_V$  is deterministic and takes as input a public key pk, a message *m*, a signature  $\hat{\sigma}$ , a (partial) attribute vector  $\vec{L'}$ , an index set  $I_{op}$  corresponding to the non- $\perp$  indices in  $\vec{L'}$ , and a proof  $\bar{\pi}$ . It outputs 1 (ACCEPT) or 0 (REJECT).
- The verification algorithm V is deterministic and takes as input a public key pk, a signature σ̂, and a message m. It outputs 1 (ACCEPT) or 0 (REJECT).

Definition 2.11 (Correctness of Blind Signatures with Attributes). We say that a blind signature scheme with attributes  $BSA = (\mathcal{G}, \mathcal{R}, \mathcal{S}, \mathcal{U}, \mathcal{V}, \mathcal{SH})$  is perfectly correct if for all public parameters pp, all vectors of attributes  $\vec{L}$ , all index sets  $I_{op} \subset [n]$ , and all messages m, it holds that

$$\Pr\left[\begin{array}{c} (\mathsf{pk},\mathsf{sk}) \leftarrow_{\$} \mathcal{G}(\mathsf{pp}) \\ (C, st_R, \pi) \leftarrow_{\$} \mathcal{R}_{\mathcal{U}}(\vec{L}) \\ 1 \leftarrow_{\$} \mathcal{R}_{\mathcal{S}}(C, \pi) \\ \mathcal{V}(\mathsf{pk}, \hat{\sigma}, m) = 1 \land \\ \mathcal{S}\mathcal{H}_V(\mathsf{pk}, m, \hat{\sigma}, \vec{L'}, I_{op}, \bar{\pi}) = 1 \end{array} : \begin{array}{c} (\mathsf{pk}, \mathsf{st}_S) \leftarrow_{\$} \mathcal{S}_1(\mathsf{sk}, C) \\ (\mathsf{e}, st_Q) \leftarrow_{\$} \mathcal{U}_1(\mathsf{pk}, R, C, m, st_R) \\ \mathcal{S} \leftarrow_{\mathcal{S}}(\mathsf{sk}, st_S, R, e) \\ (\tilde{\sigma}, st_Q) \leftarrow \mathcal{U}_2(\mathsf{pk}, st_S, R, e, S, m) \\ (\vec{L'}, \bar{\pi}) \leftarrow \mathcal{S}\mathcal{H}_U(\mathsf{pk}, st_R, m, \sigma, st_Q, I_{op}) \end{array} \right] = 1.$$

Definition 2.12 (Blindness). For the blindness definition we refer the reader to the original work of Baldimtsi and Lysyanskaya [3].

We recall the definition of strong One-More Unforgeability, where an adversary wins if it is able to output  $\ell + 1$  valid messagesignature pairs after completing only  $\ell$  signing sessions. In the setting of blind signatures with attributes, this game is augmented with a registration oracle. The registration oracle  $\mathcal R$  allows the adversary to register users with attributes of its choice. The game keeps the internal state the signer would keep regarding registered users which includes a list of the registered attribute commitments. It responds with 1 or 0 like the signer side registration algorithm. The signing oracle  $S(sk, \cdot)$  is split into two oracles  $S_1$  and  $S_2$  that share session identifiers and state such that each session opened with a call to  $S_1$  can be closed with exactly one call to  $S_2$ . It allows the adversary to request signatures for previously registered users of its choice by receiving first-round and second-round signer messages. The users can be identified by their commitments to attributes. The adversary may arbitrarily interleave signing sessions (i.e. make many calls to  $S_1$  before making the call with the same session identifier to  $S_2$ ) and also leave signing sessions open at the end of the game. The game maintains a counter for how many signing sessions were closed. It is not possible to close the same signing session twice.

Definition 2.13 (Strong One-More Unforgeability (OMUF)). For a blind signature with attributes  $BSA = (\mathcal{G}, \mathcal{R}, \mathcal{S}, \mathcal{V}, \mathcal{SH})$ , a positive integer  $\ell \in \mathbb{Z}^+$ , and an adversary  $\mathcal{A}$ , we define the game  $\ell$ -OMUF as follow:

- Setup. Generate a pair of keys (pk, sk) ← \$ G(pp), and invoke A<sup>R(.),S(sk,.)</sup>.
- Online Phase.  $\mathcal{A}$  may query its oracles  $\mathcal{R}(.), \mathcal{S}(sk, .)$  arbitrarily and in an interleaved fashion as long as it completes at most  $\ell$  sessions with  $\mathcal{S}$ .
- **Output Determination.** The game outputs 1 iff  $\mathcal{A}$  outputs  $k \ge \ell + 1$  pairwise-distinct tuples  $(\hat{\sigma}_1, m_1), \ldots, (\hat{\sigma}_k, m_k)$  such that for all  $i \in [k]$ ,  $\mathcal{V}(pk, \hat{\sigma}_i, m_i) = 1$ .

We define  $\mathcal{A}$ 's advantage in winning the game  $\ell$ -OMUF against a blind signature scheme with attributes BSA as

$$\mathrm{Adv}_{\mathsf{BSA}}^{\ell \mathsf{-}\mathsf{OMUF}}(\mathcal{A}) \coloneqq \Pr[\ell \mathsf{-}\mathsf{OMUF}_{\mathsf{BSA}}^{\mathcal{A}} = 1].$$

We say that BSA is  $(t, \varepsilon, \ell)$ -OMUF-secure if for all adversaries  $\mathcal{A}$  that run in time at most t,  $\operatorname{Adv}_{\mathsf{BSA}}^{\ell-\mathsf{OMUF}}(\mathcal{A}) \leq \varepsilon$ .

We present the definition of secure showing. Intuitively, this security definition captures that the adversary cannot re-link received signatures to other attribute vectors. Definition 2.14 (Secure Showing (from [35]))). For a blind signature with attributes BSA =  $(\mathcal{G}, \mathcal{R}, \mathcal{S}, \mathcal{V}, \mathcal{SH})$ , a positive integer  $\ell \in \mathbb{Z}^+$ , and an adversary  $\mathcal{A}$ , we define the game  $\ell$ -SH as follows.

- **Setup.** Generate key pair (pk, sk)  $\leftarrow$   $\mathcal{G}(pp)$  and invoke  $\mathcal{H}^{\mathcal{R}(\cdot), \mathcal{S}(sk, \cdot)}$ .
- Online Phase. The adversary may query its oracles arbitrarily and in an interleaved fashion as long as it completes at most *l* sessions with S. For *i* ∈ [*l*], let C'<sub>i</sub> denote the registered commitment corresponding to the *i*<sub>th</sub> session with S(sk, ·). We define C'<sub>i</sub> = ⊥ if there is no session *i* (this occurs if A makes fewer than *l* signing queries).
- **Output Determination.** The adversary outputs up to  $\ell$  pairwise distinct tuples of the form  $(m_i, \sigma_i, C_i, \vec{L}_i, r_i, \pi_i)$  as well as  $\ell$  pairs  $(\vec{L}'_i, r'_i)$ . The game outputs 1 if all of the following hold and 0 otherwise:
  - − For all  $i \in [\ell]$ , either  $r'_i$  is a valid opening of  $C'_i$  to  $\vec{L'_i}$  or  $C'_i = \bot$  (recall that  $C'_i = \bot$  when the adversary chose to open less than  $\ell$  signing sessions).
  - For all tuples  $(m_i, \sigma_i, C_i, \vec{L}_i, \pi_i)$  in the adversary's output, it holds that  $S\mathcal{H}_V(\text{pk}, \sigma_i, C_i, \vec{L}_i, \pi_i) = 1$ .
  - For all tuples  $(m_i, \sigma_i, \vec{L}_i, \pi_i)$  in the adversary's output, it holds that  $\mathcal{V}(\text{pk}, m_i, \sigma_i) = 1$ .
  - All  $\vec{L}_i, \vec{L}'_i$  contain an entry for each attribute, i.e., they do not contain the ⊥ symbol anywhere.
  - − The multiset of all attribute vectors  $\{\vec{L}_i\} \not\subseteq \{\vec{L}'_i : C'_i \neq \bot\}$ .

We define  $\mathcal{A}$ 's advantage in winning the game  $\ell$ -SH against a blind signature scheme with attributes BSA as

$$\mathrm{Adv}_{\mathrm{BSA}}^{\ell-\mathrm{SH}}(\mathcal{A}) \coloneqq \Pr[\ell-\mathrm{SH}_{\mathrm{BSA}}^{\mathcal{A}} = 1]$$

We say that BSA is  $(t, \varepsilon, \ell)$ -SH secure if for all adversaries that run in time at most t,  $\operatorname{Adv}_{BSA}^{\ell-SH}(\mathcal{A}) \leq \varepsilon$ .

#### 2.4 A Stronger Secure Showing Definition

The secure showing property as defined in [35] and presented above as definition 2.14, does not cover all possible attacks on the Secure Showing algorithm that one might consider.

In the real world, users typically do not reveal all their attributes but instead disclose only those necessary to access a specific service. For instance, a movie streaming platform might require users to present message-signature pairs (as proof of purchase) along with an opening of their age attribute to access age-restricted movies. However, users would not want to, nor are they likely required to, reveal their full attribute vector, which might include details such as their place of residence, name, or nationality.

To use signatures as tokens securely, it must not be possible to show more signatures for a given vector of attributes than were originally requested for the corresponding commitment. Consider the following example: two colluding users, one aged 18 and the other 15, request two signatures (i.e., purchase two movies) tied to a commitment containing the 18-year-old's attribute vector and one signature tied to a commitment containing the 15-year-old's attribute vector. These users should not be able to present three signatures with the age attribute set to 18—effectively allowing the 15-year-old to access age-restricted content—while still withholding other attributes, such as their names or places of residence.

This scenario is not addressed by the one-more unforgeability property, as the users output exactly as many signatures as they requested. It is also not covered by the previous definition of secure showing, since the users in that case do not reveal their full attribute vectors. To address this gap, we extend the secure showing definition to what we term *strong secure showing*. This new definition models a game where users are permitted to partially open attribute vectors using the scheme's showing algorithm and are considered successful if they execute an attack similar to the one described above.

One concern raised in [35] was that the winning criterion might be inefficient to check because the challenger would have to check that there exists no mapping from the "showed" partial vectors to the full vectors that the adversary requested signatures for. In particular, we cannot trust the adversary to output a "witness" for winning the game (i.e. a bad mapping that leaves at least one vector unmapped), as a better mapping might exist.

We resolve this through the observation that this mapping can be expressed as a matching in a bipartite graph and describe a graphbased approach for efficiently checking the winning condition in remark 2.16.

Definition 2.15 (Strong Secure Showing). For a blind signature with attributes  $BSA = (\mathcal{G}, \mathcal{R}, \mathcal{S}, \mathcal{V}, \mathcal{SH})$ , a positive integer  $\ell \in \mathbb{Z}^+$ , and an adversary  $\mathcal{A}$ , we define the game  $\ell$ -SH as follows.

- Setup. Generate key pair (pk, sk)  $\leftarrow_{\$} \mathcal{G}(pp)$  and invoke  $\mathcal{R}^{\mathcal{R}(.),\mathcal{S}(sk,.)}$ . Oracles  $\mathcal{R}(.)$  and  $\mathcal{S}(sk,.)$  share state and we require that session *i* with  $\mathcal{S}(sk,.)$  uniquely identifies some prior session *j* with  $\mathcal{R}(.)$ .
- Online Phase. The adversary may query its oracles arbitrarily and in an interleaved fashion as long as it completes at most *l* sessions with S. For *i* ∈ [*l*], let C'<sub>i</sub> denote the commitment corresponding to the *i*<sub>th</sub> session with S(sk, .). We define C'<sub>i</sub> = ⊥ if there is no session *i*.
- Output Determination. The adversary outputs up to *l* tuples of the form (*m<sub>i</sub>*, *σ<sub>i</sub>*, *C<sub>i</sub>*, *L<sub>i</sub>*, *r<sub>i</sub>*, *π<sub>i</sub>*) such that the (*m<sub>i</sub>*, *σ<sub>i</sub>*, *C<sub>i</sub>*) parts are pairwise distinct as well as *l* pairs (*L<sub>i</sub>'*, *r<sub>i</sub>'*). The game outputs 1 if all of the following hold and 0 otherwise:
  - For all *i* ∈ [*ℓ*], either  $r'_i$  is a valid opening of  $C'_i$  to  $\vec{L}'_i$  or  $C'_i = \bot$ .
  - For all tuples (m<sub>i</sub>, σ<sub>i</sub>, L<sub>i</sub>, π<sub>i</sub>) in the adversary's output, it holds that SH<sub>V</sub>(pk, σ<sub>i</sub>, L<sub>i</sub>, π<sub>i</sub>) = 1.
  - For all tuples (m<sub>i</sub>, σ<sub>i</sub>, L<sub>i</sub>, π<sub>i</sub>) in the adversary's output, it holds that V(pk, m<sub>i</sub>, σ<sub>i</sub>) = 1.
  - There exists no injective mapping

$$\mu: \{\vec{L}_i\}_{i=1}^{\ell} \to \{\vec{L}'_i\}_{i=1}^{\ell}$$

such that for all *i*, it holds that  $\vec{L}_i \subseteq \mu(\vec{L}_i)$ .

We define  $\mathcal{A}$ 's advantage in winning the game  $\ell$ -SH against a blind signature scheme with attributes BSA as

$$\mathrm{Adv}_{\mathrm{BSA}}^{\ell-\mathrm{SH}}(\mathcal{A}) \coloneqq \Pr[\ell - \mathrm{SH}_{\mathrm{BSA}}^{\mathcal{A}} = 1]$$

We say that BSA is  $(t, \varepsilon, \ell)$ -SH secure if for all adversaries that run in time at most t,  $Adv_{BSA}^{\ell-SH}(\mathcal{A}) \leq \varepsilon$ .

Remark 2.16. It is possible to check the winning condition of the above game in time  $O(\ell^{2.5})$  by using the Hopcroft-Karp[33] algorithm to find a maximum cardinality matching. To apply the algorithm, we briefly explain how to view the sets of vectors  $\{\vec{L}_i\}_{i=1}^{\ell}$  and  $\{\vec{L}'_i\}_{i=1}^{\ell}$  as a graph: Each vector from the multiset  $\{\vec{L}_i\}_{i=1}^{\ell}$  corresponds to a vertex  $v_i$  and each vector from the multiset  $\{\vec{L}'_i\}_{i=1}^{\ell}$  corcorresponds to a vertex  $w_i$  in the graph, There exists an edge between  $v_i$  and  $w_j$  if and only if  $\vec{L}_i \subset \vec{L}'_j$ .

It is easy to see that this graph is bipartite as there are only edges between  $v_i$ 's and  $w_j$ 's but not within the set of the  $v_i$ 's or within the set of the  $w_j$ 's.

The running time of the Hopcroft-Karp algorithm is  $n^{2.5}$  where *n* is the number of vertices - in our case the number of vertices is  $2\ell$ , thus the running time of the algorithm lies in  $O(\ell^{2.5})$ .

## 2.5 Double-Spending Definition

In the following, extend the definition of blind signature with attributes (Def. 2.10) to also support **double-spend protection**, by modifying the user side registration algorithm,  $\mathcal{R}_{\mathcal{U}}$ , and defining additional algorithms for double-spending detection, resulting in a type of scheme called BSADS = ( $\mathcal{G}, \mathcal{R}, \mathcal{S}, \mathcal{U}, \mathcal{VSH}, \mathcal{DSI}, \mathcal{DSV}$ ). Unless stated otherwise below, the algorithms of a BSADS have the same input and output behaviour, correctness, and security notions as BSA.

- *R*<sub>U</sub> takes as input a vector of attributes *L* where the first attribute, denoted by *ID*, is a unique identifier of the user. It outputs a commitment *C* to *L*, a registration state *st*<sub>R</sub>, and a proof π where we use the proof of knowledge of a partial opening as described in definition B.6 to reveal *ID*.
- DSI the double-spend identification algorithm takes as input the issuer's public key pk, a signature  $\sigma$ , the signer's internal state  $st_S$ , and two different presentation proofs  $\pi_1, \pi_2$  and it either outputs a user *ID* and a proof of guilt  $\pi_O$ , or  $\perp$ .
- DSV the double-spend verification algorithm takes as input a user *ID* and a proof of guilt π<sub>Q</sub> and it outputs 1 (ACCEPT) or 0 (REJECT).

Discussion. The specifics of the registration protocol will depend on the exact policy of each application scenario. For simplicity, in our definition above, we assume that the first attribute of the vector  $\vec{L}$  is a unique identifier of the user, denoted as ID, and this uniqueness is guaranteed by the attached proof  $\pi$ . For example, in certain cases, during the registration process, the user might present a physical document, such as a passport, and prove (e.g., by partially opening the commitment C) that the encoded attributes match those in their passport. In such scenarios, ID could be the unique passport number. Regardless of the specific implementation, ID must be unique for each user and should be a value that cannot be duplicated or obtained by another user to prevent framing attacks where a malicious user attempts to impersonate another by registering under their identity and engaging in fraudulent activities.

A blind signature with attributes and double-spend protection BSADS needs satisfy two additional security properties: doublespending identification and exculpability. Double-spending identification guarantees that no adversary is able to present the same token twice without having their public key revealed.

Definition 2.17 (Double-spending Identification). For a blind signature with attributes and double-spending protection BSADS =  $(\mathcal{G}, \mathcal{R}, \mathcal{S}, \mathcal{U}, \mathcal{V}, \mathcal{SH}, \mathcal{DSI}, \mathcal{DSV})$ , and an adversary  $\mathcal{A}$ , we define the game DSP as follows.

- Setup. Generate a pair of keys (pk, sk) ← \$ G(pp), and invoke A<sup>R(.),S(sk,.)</sup>.
- **Online Phase.**  $\mathcal{A}$  may query its oracles  $\mathcal{R}(.)$ ,  $\mathcal{S}(sk, .)$  arbitrarily and in an interleaved fashion.
- Output Determination. The game outputs 1 iff  $\mathcal{A}$  outputs:  $(\sigma, \vec{L'_1}, \bar{\pi_1})$  and  $(\sigma, \vec{L'_2}, \bar{\pi_2})$  such that  $\vec{L'_1} \neq \vec{L'_2} \lor \bar{\pi_1} \neq \bar{\pi_2}$  $\mathcal{SH}_V(\sigma, \vec{L'_1}, \bar{\pi_1}) = 1$  and  $\mathcal{SH}_V(\sigma, \vec{L'_2}, \bar{\pi_2}) = 1$  and  $\mathcal{DSI}(\sigma, \bar{\pi_1}, \bar{\pi_2}) = \bot$

We define  $\mathcal R$  's advantage in winning the game DSP against a blind signature scheme with attributes BSADS as

$$\operatorname{Adv}_{\operatorname{BSADS}}^{\operatorname{DSP}}(\mathcal{A}) \coloneqq \Pr[\operatorname{DSP}_{\operatorname{BSADS}}^{\mathcal{A}} = 1].$$

We say that BSADS is  $(t, \varepsilon)$ -DSP-secure if for all adversaries  $\mathcal{A}$  that run in time at most t,  $\operatorname{Adv}_{BSADS}^{DSP}(\mathcal{A}) \leq \varepsilon$ .

Finally, we also define exculpability which guarantees that a malicious signer cannot wrongly accuse an honest user of double-spending.

Definition 2.18 (Exculpability). For a blind signature with attributes and double-spending protection BSADS = ( $\mathcal{G}, \mathcal{R}, \mathcal{S}, \mathcal{U}, \mathcal{V}, \mathcal{SH}, \mathcal{DSI}, \mathcal{DSV}$ ), and an adversary  $\mathcal{A}$ , we define the game Exc as follows.

- **Setup.**  $\mathcal{A}$  outputs pk and is given access to honest user oracles  $\mathcal{A}^{\mathcal{R}_{\mathcal{U}}(.),\mathcal{U}(.),\mathcal{SH}_{U}(.)}$ . Initialize  $Q_{U}$  to hold the set of honest user *ID*'s generated by the oracle calls.
- **Online Phase.**  $\mathcal{A}$  is may query the following oracles in an interleaved fashion.
  - **Registration Oracle**  $\mathcal{R}_{\mathcal{U}}(.)$  It takes as input an attributes vector  $\vec{L}$  with a unique first attribute (recall that *ID* is the first attribute) and runs the user's attributes registration algorithm  $\mathcal{R}_{\mathcal{U}}$  on it to obtain  $C, \pi, st_{\mathcal{R}}$ . It outputs  $C, \pi$  to the malicious signer.
  - User Oracles  $\mathcal{U}(.)$  For the first round, it takes as input a commitment *C* that was an output of a previous registration query, a first signer commitment *R*, and a message *m*. It runs the  $\mathcal{U}_1$  algorithm and returns a challenge *e*. For the second round, it takes as input a signer response *S*, runs the  $\mathcal{U}_2$  algorithm, and returns the resulting signature  $\sigma$  on the message (or  $\perp$  if  $\mathcal{U}_2$ did not result in a signature). It keeps the showing state  $st_Q$ .
  - **Showing Oracle**  $SH_U(.)$  It takes as input a signature  $\sigma$  from the outputs of the user oracles and a set  $I_{op}$  of indices to open. If the signature has never been queried to the showing oracle before, it runs the showing algorithm  $SH_U$  for this signature using the internal states  $st_R$  and  $st_O$  it kept from registration and signature generation.

**Output Determination.** The game outputs 1 iff  $\mathcal{A}$  outputs: (*ID*,  $\pi_Q$ ) such that  $ID \in Q_U$  and  $\mathcal{DSV}(ID, \pi_Q) = 1$ .

We define  $\mathcal A$  's advantage in winning the game Exc against a blind signature scheme with attributes BSADS as

$$\operatorname{Adv}_{\operatorname{BSADS}}^{\operatorname{Exc}}(\mathcal{A}) \coloneqq \Pr[\operatorname{Exc}_{\operatorname{BSADS}}^{\mathcal{A}} = 1]$$

We say that BSADS is  $(t, \varepsilon)$ -Exc-secure if for all adversaries  $\mathcal{A}$  that run in time at most t,  $Adv_{BSADS}^{Exc}(\mathcal{A}) \leq \varepsilon$ .

# **3 ANONYMOUS CREDENTIALS LIGHT**

We now present Anonymous Credentials light scheme of [3] along with protocols for partial attribute showing and double spending detection.

#### 3.1 The Main ACL Construction

We first recall the Anonymous Credentials Light scheme from [3]. For the reader's convenience, we have included a figure of the signing interaction in fig. 3 in appendix D.

For a group  $\mathbb{G}$  with generator g and prime order p described by parameters  $pp_{\mathbb{G}}$  and hash functions  $H_{pp}, H_1: \{0, 1\}^* \rightarrow \mathbb{G}$ ,  $H_{reg}, H_{sh}, H_3: \{0, 1\}^* \rightarrow \mathbb{Z}_p$  modeled as random oracles, the scheme ACL =  $(\mathcal{G}, \mathcal{R}, \mathcal{S}, \mathcal{U}, \mathcal{V})$  is defined as follows:

- $\mathcal{G}(\operatorname{pp}_{\mathbb{G}})$ : Sample  $x \leftarrow_{\$} \mathbb{Z}_p, h \leftarrow_{\$} \mathbb{G}$ , compute  $y \coloneqq g^x$ , generate  $z \coloneqq H_1(g, h, y)$ , set sk  $\coloneqq x$  and pk  $\coloneqq (g, h, y, z)$ , and output (pk, sk).
- $\mathcal{R}_{\mathcal{U}}(\vec{L})$ : Parse  $(L_1, \ldots, L_n) = \vec{L}$ , obtain  $h_0, \ldots, h_n$  by querying  $0, \ldots, n$  to  $\mathsf{H}_{pp}$ , sample  $r_{com} \leftarrow_{\$} \mathbb{Z}_p$  and compute  $C := h_0^{r_{com}} \cdot h_1^{L_1} \cdot \ldots \cdot h_n^{L_n}$ . Compute a proof of knowledge  $\pi$  of an opening of C, using the Fiat-Shamir transform of the protocol described in definition 2.7 with random oracle  $\mathsf{H}_{reg}$  and output  $(C, st_{\mathcal{R}}, \pi)$ .
- $\mathcal{R}_{\mathcal{S}}(C, \pi)$ : Verify that  $\pi$  is a proof of knowledge of an opening of *C* using the verification of the protocol described in definition 2.7. Output 1 (ACCEPT) if yes, 0 (REJECT) otherwise.
- S: The signer algorithm. It consists of  $S_1$  and  $S_2$ :
  - $S_1(\mathsf{sk}, C): \text{Sample } d, s_1, s_2, u, \text{rnd} \leftarrow_{\$} \mathbb{Z}_p, \text{ generate } z_1 \\ \coloneqq g^{\mathsf{rnd}} \cdot C, z_2 \coloneqq z/z_1, \text{ compute } a \coloneqq g^u, b_1 \coloneqq g^{s_1} \cdot z_1^d, \\ b_2 \coloneqq h^{s_2} \cdot z_2^d, \text{ set } st_S \coloneqq (d, s_1, s_2, u), R \coloneqq (\mathsf{rnd}, a, b_1, b_2), \text{ and output } (R, st_S).$
  - $S_2(\operatorname{sk}, \operatorname{st}_S, R, e)$ : Parse  $d, s_1, s_2, u = \operatorname{st}_S, x := \operatorname{sk}$ , compute  $c := e d \mod q$  and  $r := u c \cdot x \mod q$ , and return  $S := (c, d, r, s_1, s_2)$ .
- $\mathcal{U}$ : The user algorithm consists of two algorithms  $\mathcal{U}_1$  and  $\mathcal{U}_2$ :
  - $\mathcal{U}_1(\operatorname{pk}, R, m, C)$ : Parse  $(\operatorname{rnd}, a, b_1, b_2) \leftarrow R$ , sample  $\gamma \leftarrow_{\$} \mathbb{Z}_p^*, \tau, t_1, t_2, t_3, t_4, t_5 \leftarrow_{\$} \mathbb{Z}_p$ , generate  $z_1 \coloneqq g^{\operatorname{rnd}} \cdot C$ ,  $\zeta \coloneqq z^{\gamma}, \zeta_1 \coloneqq z_1^{\gamma}, \zeta_2 \coloneqq \zeta/\zeta_1$ , compute  $\alpha \coloneqq a \cdot g^{t_1} \cdot y^{t_2}, \beta_1 \coloneqq b_1^{\gamma} \cdot g^{t_3} \cdot \zeta_1^{t_4}, \beta_2 \coloneqq b_2^{\gamma} \cdot h^{t_5} \cdot \zeta_2^{t_4}, \eta \coloneqq z^{\tau}$ , generate  $\varepsilon \coloneqq H_3(\zeta, \zeta_1, \alpha, \beta_1, \beta_2, \eta, m)$ , compute  $e \coloneqq \varepsilon - t_2 - t_4$ mod q, set  $st_{\mathcal{U}} \coloneqq (\gamma, \tau, t_1, t_2, t_3, t_4, t_5, \operatorname{rnd})$ , and return  $(e, st_{\mathcal{U}})$ .
  - $\mathcal{U}_2(\text{pk}, st_{\mathcal{U}}, R, e, S, m)$ : Parse  $(\gamma, \tau, t_1, t_2, t_3, t_4, t_5) = st_{\mathcal{U}}$ , and  $(c, d, r, s_1, s_2, \text{rnd}_1, \text{rnd}_2) = S$ , compute  $\rho := r + t_1, \omega := c + t_2, \sigma_1 := \gamma \cdot s_1 + t_3, \sigma_2 := \gamma \cdot s_2 + s_3$

 $t_{5}, \delta \coloneqq d + t_{4}, \mu \coloneqq \tau - \delta \cdot \gamma, \text{ and return the signature} \\ \hat{\sigma} \coloneqq (\zeta, \zeta_{1}, \rho, \omega, \sigma_{1}, \sigma_{2}, \delta, \mu) \text{ if } \omega + \delta \equiv \mathsf{H}_{3}(\zeta, \zeta_{1}, g^{\rho} \cdot y^{\omega}, g^{\sigma_{1}} \cdot \zeta_{1}^{\delta}, h^{\sigma_{2}} \cdot \zeta_{2}^{\delta}, z^{\mu} \cdot \zeta^{\delta}, m), \perp \text{ otherwise. Output} \\ st_{O} = (\gamma, \text{rnd}).$ 

•  $\mathcal{V}(\mathsf{pk}, m, \hat{\sigma})$ : Parse  $\hat{\sigma}$  as  $(\zeta, \zeta_1, \rho, \omega, \sigma_1, \sigma_2, \delta, \mu)$ , and output 1 if  $\zeta \not\equiv 1$  and  $\omega + \delta \equiv \mathsf{H}_3(\zeta, \zeta_1, g^{\rho} \cdot y^{\omega}, g^{\sigma_1} \cdot \zeta_1^{\delta}, h^{\sigma_2} \cdot (\zeta/\zeta_1)^{\delta}, z^{\mu} \cdot \zeta^{\delta})$ , 0 otherwise.

# 3.2 ACL Showing of Partial Attributes

Showing and verification of showing is depicted in fig. 1. We note that it is also possible to "show" a signature without revealing any attributes, i.e. when  $I_{op}$  is empty. In this case, the user can run a proof of same discrete logarithm (see definition B.1) for the blinded commitment parameters, as well as a proof of knowledge of an opening of a Pedersen commitment to the blinded parameters like the one used at registration.

The procedure described in fig. 1 allows for extraction in the AGM:

THEOREM 3.1 (EXTRACTOR FOR SHOWING PROCEDURE). For any algebraic adversary  $\mathcal{U}$  that, on input of commitment parameters  $pp_{GPC} = \vec{h}$ , it outputs an ACL public key pk along with a tuple  $(m, \sigma, \vec{L}', I_{op}, \bar{\pi})$  such that  $S\mathcal{H}_V(pk, m, \sigma, \vec{L}', I_{op}, \bar{\pi}) = 1$ , there exists an extractor  $\mathcal{E}$  that takes  $\mathcal{U}$ 's output  $(pk, (m, \sigma, \vec{L}', I_{op}, \bar{\pi}))$  along with all algebraic representations  $\mathcal{U}$  outputs as input, and it outputs a witness  $\gamma$  of the proof of same discrete logarithm with probability  $1 - Q_h \frac{1+n}{p}$  (over the choice of the random oracle) and (if the former happens) additionally either a witness  $\vec{L}$  which is a partial opening to the commitment  $\zeta_1$  on the indices  $i \notin I$ , or a solution  $R, \vec{z}, \vec{z}'$  to the **BINDING** game of Pedersen commitments with probability  $1 - \frac{2}{p}$  over the choice of the random oracle.

We prove this in appendix B.

#### 3.3 Secure Showing of ACL

THEOREM 3.2. If the discrete logarithm problem is  $(t, \varepsilon)$ -hard in the group  $\mathbb{G}$ , then ACL is  $(t', \varepsilon')$ -SSsecure in the AGM + ROM with  $t' \approx t$  and

$$\varepsilon' \le (n+21) \cdot \varepsilon + \frac{(n+3) \cdot \ell \cdot Q_H + 52 + 4 \cdot Q_C}{p} + \frac{1}{n},$$

where  $\ell$  is the number of signing sessions closed,  $Q_H$  is the number of hash queries the adversary made to  $H_{sh}$ ,  $Q_C$  is the number of commitments  $\mathcal{A}$  registers, and n is the maximal number of attributes contained in each attribute vector.

PROOF. We work in the AGM and show that an adversary that breaks the secure showing property can be used to break the discrete logarithm assumption in the underlying group. Our proof technique varies depending on the adversary's strategy for breaking the secure showing property - namely it can either provide a set of signatures where for some signatures, the blinded  $\zeta_1$  components do not match the  $C \cdot g^{\text{rnd}}$  components of the signing sessions. This can happen either in the form of one "odd one out" signature that matches no session (we rule this out in G<sub>2</sub>), or in two signatures that match the same signing session (we rule this out in G<sub>3</sub>). The last possible strategy is for the adversary to break the binding property of the

$SH_U(\text{pk}, st_R = (r_{com}, L), m, \sigma, st_O = (\text{rnd}, \gamma), I_{op},)$	$SH_V(\text{pk}, \sigma, m, L', I_{op}, \bar{\pi} = (\pi_{sdl}, \pi_{op}))$
	If $\mathcal{V}(pk, \sigma, \zeta_1, m) = 0$
$\Gamma := g^{\gamma}$	or $\exists i \in I_{op} : L'_i = \bot \lor \exists i \notin I_{op} : L'_i \neq \bot$
$\vec{\Psi} \coloneqq \{\psi_k \coloneqq h_k^{\gamma}\}_{k \in \llbracket n \rrbracket}$	Return 0
$r_{sdl} \leftarrow_{\$} \mathbb{Z}_p$	$(\Psi, \Gamma, g_{sdl}, z_{sdl}, s_{sdl}, h_{sdl}) \leftarrow \pi_{sdl}$
$\vec{h}_{sdl} \coloneqq \{h_{sdl,k} \coloneqq h_k^{r_{sdl}}\}_{k \in \llbracket n \rrbracket}$	$(\psi_1,\ldots,\psi_n) \leftarrow \Psi$
$z_{sdl} \coloneqq z^{r_{sdl}}$	$(h_{sdl,1},\ldots,h_{sdl,n}) \leftarrow h_{sdl}$
$g_{sdl} \coloneqq g^{r_{sdl}}$	$c \leftarrow_{\$} H_{sh}(pk, \sigma, \zeta_1, m, \Psi, \Gamma, g_{sdl}, z_{sdl}, \vec{h}_{sdl}, R, \vec{L'})$
$r_g, \{r_i\}_{i \in [n]_0, \setminus I_{op}} \leftarrow_{\$} \mathbb{Z}_p$	If $\Gamma^c \cdot g^{s_{sdl}} \neq g_{sdl}$ or $\zeta^c \cdot z^{s_{sdl}} \neq z_{sdl}$
$R \coloneqq g^{r_g} \cdot \prod_{i \in [n] \setminus I_{op}} \psi_i^{r_i}$	or $\bigvee_{i \in [n]} \psi_i^c \cdot h_i^{s_{sdl}} \neq h_{sdl,i}$
for $i \in [n] \setminus I_{op} : L'_i := \bot$	Return 0
for $i \in I_{op}$ : $L'_i := L_i$	$(R,S) \leftarrow \pi_{op}$
$c \leftarrow_{\$} H_{sh}(pk,\sigma,\sigma.\zeta_1,m,\vec{\Psi},\Gamma,g_{sdl},z_{sdl},\vec{h}_{sdl},R,\vec{L'})$	$\zeta_1' \coloneqq \zeta_1 / \prod_{i \in I_{op}} \psi_i^{L_i'}$
$s_{sdl} \coloneqq r_{sdl} - c \cdot \gamma$	If $\zeta_1^{\prime c} \cdot \Gamma^{s_{\Gamma}} \cdot \prod_{i \in [n]_0 \setminus I_{op}} \psi_i^{s_i} \neq R$
$s_0 \coloneqq r_0 - c \cdot r_{com}$	Return 0
$\{s_i \coloneqq r_i - L_i \cdot c\}_{i \in [n]_0 \setminus I_{op}}$	Return 1
$s_{\Gamma} \coloneqq r_q - \operatorname{rnd} \cdot c$	
$S \coloneqq \{\tilde{s_i}\}_{k \in I_{op}} \cup \{s_{\Gamma}\}$	
$\pi_{sdl} \coloneqq (\Psi, \Gamma, g_{sdl}, z_{sdl}, s_{sdl}, h_{sdl})$	
$\pi_{op} := (R, S)$	
Output $\vec{L}'$ , $(\pi_{sdl}, \pi_{op})$	
*	

Figure 1: The Showing procedure. The  $\zeta_1$  component of the signature  $\sigma$ , denoted here by  $\sigma.\zeta_1$  is a blinded commitment C, for each a signature  $\sigma$  was issued via an interaction with an honest signer. ( $\gamma$ , rnd) is the blinding factor such that  $\zeta_1 = (g^{rnd} \cdot C)^{\gamma}$ . The proof consists of a part that proves the correct blinding of the commitment parameters via a proof of same discrete logarithm and of a part that proves knowledge of an opening for the remaining part of the commitment that is not revealed by the set of opened indices  $I_{op}$ .

underlying Pedersen commitment scheme which we rule out in lemma 3.9.

Game G1. This is the secure showing game for ACL.

*Game* G<sub>2</sub>. This game aborts if the following happens: The adversary outputs at least one signature  $\sigma^*$  such that the commitment  $\zeta^*, \zeta_1^*$  does not "match" any of the session tags on the signer's side. This means, there exists no  $z_{1,i}$  such that  $d\log_z \zeta^* = d\log_{z_{1,i}} \zeta_1^*$ .

We will show that this contradicts the so-called *restrictive blind-ing lemma*, see lemma 3.4. We prove this lemma using mostly techniques from [35] to show that an adversary can be used to break DLP if it wins this game.

First, we describe the restrictive blinding game:

Definition 3.3 (Restrictive Blinding Game). We describe the restrictive blinding game for an adversary  $\mathcal{A}$ :

- **Setup:** The challenger samples a key pair (pk, sk) according to  $\mathcal{G}(pp_{\mathbb{G}})$ . This uses the random oracle H<sub>1</sub> which allows the challenger to keep track of the discrete logarithms of its outputs. It outputs pk to the adversary.
- **Online Phase:** The adversary gets access to the following oracles
- **Oracle** H<sub>1</sub>: lazily samples  $w \leftarrow_{\$} \mathbb{Z}_p$  and returns  $g^w$ . Keeps a list of the hash inputs, w, and the output  $g^w$ .
- **Oracle**  $H_{pp}$ : lazily samples  $v \leftarrow_{\$} \mathbb{Z}_p$  and returns  $g^v$ . Keeps a list of the hash inputs v, and the output  $g^v$ .

**Oracle** H<sub>3</sub>: lazily samples  $\varepsilon \leftarrow_{\$} \mathbb{Z}_p$  and returns  $\varepsilon$ 

 $\mathcal{R}_{\mathcal{S}}$ : Register new commitments and store them in list to ensure all commitments coming in  $\mathcal{S}_1$  queries were already registered.

- $S_1$ : runs the signing procedure  $S_1$  of ACL and returns its output along with a fresh session ID. Keeps a list of the commitment *C* and the random value rnd.
- $S_2$ : takes as input a session ID and a challenge *e*. If the session ID has never been used before in a  $S_2$  query, it runs the  $S_2$  procedure of ACL using the internal signer state corresponding to the session ID with *e* and returns its output.
- **Output Determination:**  $\mathcal{A}$  submits a list of message-sign ature pairs  $(m_i, \sigma_i)$  for  $i = 1..\ell$ . The game outputs 1 if there is a pair  $(m, \sigma = (\zeta, \zeta_1, ...))$  in the list such that the signature is valid on the message m under the public key pk and it holds that there exists no closed signing session sid such that  $dlog_z \zeta = dlog_{z_1 sid} \zeta_1$ .

We define the advantage of the adversary in the game RB as

$$\operatorname{Adv}_{\operatorname{ACL}}^{\operatorname{RB}}(\mathcal{A}) \coloneqq \Pr[\operatorname{RB}_{\operatorname{ACL}}^{\mathcal{A}} = 1].$$

LEMMA 3.4 (RESTRICTIVE BLINDING). In the ROM and AGM, if DLP is  $(t, \varepsilon)$ -hard in  $\mathbb{G}$ , it holds that for any adversary running in time at most t, the advantage is at most

$$\operatorname{Adv}_{\operatorname{ACL}}^{\operatorname{RB}}(\mathcal{A}) \le 13 \cdot \varepsilon + \frac{4 \cdot Q_C + 31}{p} + \frac{1}{n}$$

where  $Q_C$  is the number of registration queries made by the adversary.

The proof can be found in appendix C

COROLLARY 3.5.

$$\Pr[G_1 = 1] \ge \Pr[G_2 = 1] + 13 \cdot \varepsilon + \frac{4 \cdot Q_C + 31}{p} + \frac{1}{n}.$$

PROOF. We provide a reduction: The reduction simulates the secure showing game to the adversary by forwarding its queries to

Foteini Baldimtsi, Julia Kastner, Julian Loss, and Omar Renawi

all oracles and their responses. The reduction then checks whether the adversary has won the secure showing game. If yes, it outputs the messages and signatures from the adversary's output to the RB game. It thus wins whenever the abort condition of  $G_2$  occurs.  $\Box$ 

*Game* **G**<sub>3</sub>. In this game, we abort if the adversary does the following: The adversary outputs at least two signatures  $\sigma'$ ,  $\sigma''$  such that the commitments  $\zeta', \zeta'_1$  and  $\zeta'', \zeta''_1$  map to the same signing session, i.e. there exists a single  $z_{1,i}$  such that both  $dlog_z\zeta' = dlog_{z_{1,i}}\zeta'_1$  and  $dlog_z\zeta'' = dlog_{z_{1,i}}\zeta'_1$ . We will show that this contradicts what we call the 2-to-1-lemma, see lemma 3.7 - this lemma can be derived from techniques used in the OMUF-proof of [35].

We begin by describing a game that formalizes the above criterion more:

*Definition 3.6 (2-to-1 Game).* We describe the *2-to-1* game  $\ell$ -TTO for an adversary  $\mathcal{A}$ :

- **Setup:** The challenger samples a key pair (pk, sk) according to  $\mathcal{G}(pp_{\mathbb{G}})$ . This uses the random oracle H<sub>1</sub> which allows the challenger to keep track of the discrete logarithms of its outputs. It outputs pk to the adversary.
- **Online Phase:** The adversary gets access to the following oracles
- **Oracle** H<sub>1</sub>: lazily samples  $w \leftarrow_{\$} \mathbb{Z}_p$  and returns  $g^w$ . Keeps a list of the hash inputs, w, and the output  $g^w$ .
- **Oracle**  $H_{pp}$ : lazily samples  $v \leftarrow_{\$} \mathbb{Z}_p$  and returns  $g^v$ . Keeps a list of the hash inputs v, and the output  $g^v$ .

**Oracle** H<sub>3</sub> lazily samples  $\varepsilon \leftarrow_{\$} \mathbb{Z}_p$  and returns  $\varepsilon$ 

- $S_1$ : runs the signing procedure  $S_1$  of ACL and returns its output along with a fresh session ID. Keeps a list of the commitment *C* and the
- $S_2$ : takes as input a session ID and a challenge *e*. If the session ID has never been used before in a  $S_2$  query, it runs the  $S_2$  procedure of ACL using the internal signer state corresponding to the session ID with *e* and returns its output.
- **Output Determination:** The adversary submits a list of message-signature pairs  $(m_i, \sigma_i)$  for  $i = 1..\ell$ . The game outputs 1 if there are two messages  $m_1, m_2$  and two signatures  $(\zeta^{(1)}, \zeta_1^{(1)}, \rho^{(1)}, \omega^{(1)}, \sigma_1^{(1)}, \sigma_2^{(1)}, \mu^{(1)}, \delta^{(1)})$  and  $(\zeta^{(2)}, \zeta_1^{(2)}, \rho^{(2)}, \omega^{(2)}, \sigma_1^{(2)}, \sigma_2^{(2)}, \mu^{(2)}, \delta^{(2)})$  im the list such that the signatures are valid on the respective messages under the public key pk and it holds that there exists a single closed signing session such that  $dlog_z \zeta^{(1)} = dlog_{C\cdot g^{rnd}} \zeta_1^{(1)}$  and  $dlog_z \zeta^{(2)} = dlog_{C\cdot g^{rnd}} \zeta_1^{(2)}$ .

We define the advantage of the adversary in the game  $\ell\text{-}\mathrm{TTO}$  as

$$\operatorname{Adv}_{\operatorname{ACL}}^{\ell-\operatorname{TTO}}(\mathcal{A}) \coloneqq \Pr[\ell\operatorname{-}\operatorname{TTO}_{\operatorname{ACL}}^{\mathcal{A}} = 1]$$

We show that this game is hard to win for an adversary:

LEMMA 3.7 (2-TO-1 BLINDING). In the ROM and AGM, if DLP is  $(t, \varepsilon)$ -hard in  $\mathbb{G}$ , it holds that for any adversary running in time at most t, the advantage is at most

$$\operatorname{Adv}_{\operatorname{ACL}}^{\ell\operatorname{-TTO}}(\mathcal{A}) \leq 7 \cdot \varepsilon + \frac{20}{p}.$$

The proof can be found in appendix C

We now apply this to the current game hop.

COROLLARY 3.8.

$$\Pr[G_2 = 1] \le \Pr[G_3 = 1] + (7\varepsilon + \frac{20}{p})$$

PROOF. Again, the reduction forwards all queries from the adversary to the  $\ell$ -TTO and also forwards the responses. When the adversary outputs a solution, it checks that the adversary has won the secure showing game, and then outputs the messages and signatures from the adversary's output to the  $\ell$ -TTO game. It thus wins whenever the abort condition of G<sub>3</sub> occurs.

LEMMA 3.9 (PARTIAL OPENING). In the ROM and AGM, if DLP is  $(t, \varepsilon)$ -hard in  $\mathbb{G}$ , it holds that for any adversary  $\mathcal{A}$  that runs in time at most t, the probability of winning  $G_3$  is at most:

$$\Pr[\mathbf{G}_{3}^{\mathcal{A}}] \le (n+1) \cdot \varepsilon + \frac{(n+3) \cdot \ell \cdot Q_{\mathsf{H}_{\mathsf{sh}}} + 1}{p}$$

where  $\ell$  is the number of signing sessions closed by the adversary  $\mathcal{A}$  and  $Q_{H_{sh}}$  is the number of hash queries the adversary makes to the random oracle  $H_{sh}$ 

**PROOF.** We construct an adversary  $\mathcal{R}_0$  against the binding property of the Pedersen commitment *C*. The adversary simulates the signing procedure to the adversary using the *y*-side key like an honest signer would, and extracts openings for the commitments made by the adversary. We describe this below in more details.

First of all, the reduction constructs a wrapper  $\mathcal{B}$  around the adversary which takes as input the commitment parameters and has access to the random oracle  $H_{sh}$ .

- **Setup.** The wrapper  $\mathcal{B}$  takes as input the commitment parameters  $g, \vec{h}$  for the Pedersen commitment. It samples a key for ACL using the  $\mathcal{G}(pp_{\mathbb{G}})$  algorithm while programming the RO H<sub>1</sub> so that it knows the discrete logarithms of its outputs (i.e. including *z*). It outputs the key to the adversary  $\mathcal{A}$  and programs the random oracle H<sub>pp</sub> to output  $h_i$  when queried on *i* for  $i \in [n]$ .
- Online Phase The wrapper simulates the secure showing game to the adversary by lazy-sampling all random oracles (in case of group elements with known discrete logarithm to base q) and by using the secret key to answer signing queries like an honest signer would. Whenever the adversary makes a query to a random oracle that contains a group element, the wrapper  $\mathcal{B}$  takes a note of the algebraic representations submitted for that group element. It computes a reduced representation where it replaces all occurrences of  $y, z, h, a_i, b_{1,i}, b_{2,i}$  with their respective discrete logarithms to base q, h, concretely, replace y by  $q^x, z$ by  $g^{\text{dlog}z}$  (known from programming H<sub>1</sub>), *h* by dlog*h*, *a<sub>i</sub>* by  $g^{u_i}$ ,  $b_{1,i}$  by  $g^{s_{1,i}+d_i \cdot \text{rnd}_i} \cdot C_i^{d_i}$ ,  $b_{2,i}$  by  $g^{s_{2,i} \cdot \text{dlog}h} \cdot (g^{\text{dlog}z}/(g^{\text{rnd}_i} \cdot g^{\text{dlog}z}))$  $(C_i)^{d_i}$  where  $C_i$  is recursively replaced by replacing the values in its representation. Note that by the time  $b_{1,i}, b_{2,i}$  is output, the representation of  $C_i$  is already fixed and thus there exist no circular dependencies.
- **Output determination** The wrapper outputs whatever the adversary outputs, that is the public key pk, a list of tuples  $\{(m_i, \sigma_i, \zeta_{1,i}, L_i, \pi_i)\}_{i \in k}$  and tuples  $\{(L'_i, r_i)\}_{i \in [\ell]}$ , alongside the algebraic representations that the adversary output the

first time it output a specific group element (i.e. if the group element was queried to a random oracle before, the wrapper  $\mathcal{B}$  outputs the representation that was submitted along with the RO query).

The reduction now employs the extractor  $\mathcal{E}$  from theorem 3.1 of  $\Pi$  for each of the proofs  $\pi_i = (\pi_{i,sdl}, \pi_{i,op})$  along with the commitments and algebraic representations that  $\mathcal{B}$  has output. Note that  $\mathcal{B}$  is a valid adversary that behaves like required in theorem 3.1. With probability at least  $(1 - \frac{\ell \cdot Q_{\mathsf{H}_{\mathsf{sh}}} \cdot (n+1)}{p})(1 - \frac{\ell \cdot Q_{\mathsf{H}_{\mathsf{sh}}} \cdot 2}{p}) \geq 1 - \frac{(n+3) \cdot \ell \cdot Q_{\mathsf{H}_{\mathsf{sh}}} + 1}{p}$ , the extractor outputs witnesses of the form  $\gamma_i$ , and either a commitment R with two openings  $\vec{Z}, \vec{Z}'$ , or partial openings of the commitments  $\zeta_{1,i}$  of the form  $\mathrm{rnd}_i, r_{i,0}, r_{i,j} \forall j$  s.t.  $L_{i,j} = \bot$ .

If there is one proof that yielded a commitment *R* and two openings  $\vec{Z}, \vec{Z}'$ , the reduction outputs these as its solution to the binding game. Otherwise, the reduction proceeds as follows.

As part of the output of the adversary, the reduction obtains the (full) opening vectors  $\vec{L}'_i$  for each commitment  $C_i$  that was used during registration.

The reduction now picks a session-signature-pair (rnd, *C*), (*m*,  $\hat{\sigma}$ ,  $\gamma$ ) with the following property:

$$\mathrm{dlog}_{C \cdot a^{\mathrm{rnd}}} \zeta_1 = \mathrm{dlog}_z \zeta$$

and  $\vec{L} \not\subseteq \vec{L}'$  where  $\vec{L}$  is the vector that  $\mathcal{A}$  partially opened in the showing algorithm, and  $\vec{L}'$  is the vector of the signing session that is matched through  $\gamma$ .

It unites the partial vector  $\vec{L}$  with the corresponding partial opening  $\operatorname{rnd}_i, r_{i,0}, r_{i,j} \forall j$  s.t.  $L_{i,j} = \bot$  obtained from the extractor into an opening  $\vec{L}''_i$ .

The reduction now outputs the commitment *C* along with the two openings  $\vec{L}'_i$  and  $\vec{L}''_i$  to the challenger.

Thus, the reduction wins the binding game for the Pedersen commitment scheme with probability of at least  $\Pr[\mathbf{G}_3^{\mathcal{A}} = 1] \cdot (1 - \frac{(n+3) \cdot \ell \cdot \mathcal{Q}_{\mathsf{H}_{\mathsf{sh}}} + 1}{p}) \geq \Pr[\mathbf{G}_3^{\mathcal{A}} = 1] - \frac{(n+3) \cdot \ell \cdot \mathcal{Q}_{\mathsf{H}_{\mathsf{sh}}} + 1}{p}$ . Putting this together with lemma 2.9 yields the statement.

Putting everything together we get that

$$\varepsilon' \le (n+21) \cdot \varepsilon + \frac{(n+3) \cdot \ell \cdot Q_H + 52 + 4 \cdot Q_C}{p} + \frac{1}{n}.$$

### 3.4 ACL Double-Spending Detection

We now explain how the ACL protocol can also support Double-Spending detection. We require the following modifications:

In the main protocol, as described in Fig. 3, we add the following *additional* steps to the User side of the protocol:
(1) Pick τ<sub>2</sub> ←<sub>\$</sub> ℤ<sub>p</sub>, (2) compute η<sub>2</sub> := z<sup>τ<sub>2</sub></sup>, (3) include η<sub>2</sub> in the hash computation of ε. The signature verification algorithm can either be modified to include the element η<sub>2</sub> in the hash computation, or one can treat η<sub>2</sub> as if it was part of the message when verifying signatures which enables using the verification algorithm of the plain scheme.

- In the showing protocol, as described in Fig. 1, we add the following steps: (1) The user includes  $V_{\text{name}}$  and timestamp as part of the hash computation in *c*. (2) The user computes  $\mu_2 = \tau_2 c\gamma$  and outputs  $\mu_2$  along with  $(\pi_{sdl}, \pi_{op})$ . (3) The verifier runs an updated signature verification algorithm which also includes  $z^{\mu_2} \cdot \zeta^c$  as part of H<sub>3</sub> (which verifies with  $\eta_2$  part of  $\varepsilon$  in the signature).
- DSV: On input of an ID and a double spending proof, the DSV algorithm outputs 1 iff both showings of the signature are valid, the proof of partial opening of C to ID is valid, and it holds that (C ⋅ g<sup>rnd</sup>)<sup>Y</sup> = ζ<sub>1</sub>.

*Discussion.* We note that double-spending detection for ACL was already discussed in [32] (a work that implements [3] as an e-cash scheme), but it was done in a slightly different way and without a formal security proof. Namely, in [32] the value  $\mu$  is omitted from the signature<sup>8</sup> instead of adding the additional  $\tau_2$ ,  $\eta_2$  as we did above. Then, the showing protocol uses  $\tau$  to compute  $\mu_2$ . It is critical that  $\mu$  is removed from the signature, since otherwise a single credential showing already links it to a signing session because the signer can use  $\frac{\mu_2 - \mu}{\delta - c}$  to compute  $\gamma$  and then check which  $z_1^{\gamma} = \zeta_1$ . While this double-spending protocol for ACL is correct, it does

While this double-spending protocol for ACL is correct, it does not compose well with the existing security proofs for ACL. Namely, removing  $\mu$  from the ACL signature removes the guarantees provided by existing proofs of one-more unforgeability [3, 35] as the signatures can no longer be verified on their own. Thus, we went for a modular approach, that does not affect the ACL proof, by computing additional elements ( $\tau_2$ ,  $\eta_2$ ) of the same form as ( $\tau$ ,  $\eta$ ) and adding  $\eta_2$  to the hash during the signature generation in order to use this for the double spending detection. As noted above, this modification does not affect the ACL verification algorithm and thus the ACL security proofs still hold.

LEMMA 3.10. With the modification described above, for any adversary  $\mathcal{A}$  that makes at most  $Q_{H_{sh}}$  queries to the random oracle  $H_{sh}$ , it holds that

$$\operatorname{Adv}_{\operatorname{BSADS}}^{\operatorname{DSP}}(\mathcal{A}) \leq \frac{Q_{\operatorname{H_{sh}}}^2}{p}$$

**PROOF.** Intuitively, the only way for the adversary to avoid being identified when double spending is to provoke a hash collision such that  $c \neq c'$ . We show this formally below using some game hops

*Game*  $G_1$ . This is the original DSP-game.

*Game*  $G_2$ . This game is identical to  $G_1$  except that it aborts if there is a hash collision in the random oracle *H*, i.e. two different queries by the adversary that lead to different outcomes.

<sup>&</sup>lt;sup>8</sup>This approach was also used by Abe for its blind signature scheme [1], on which ACL was based.

It is easy to see that  $\left| \Pr[\mathbf{G}_2^{\mathcal{A}} = 1] - \Pr[\mathbf{G}_1^{\mathcal{A}} = 1] \right| \leq \frac{Q_H^2}{p}$ .

We now show that the adversary  $\mathcal{A}$  has success probability 0 in  $G_2$ .

Since there is no hash collision, the two hashes *c* and *c'* are different. It thus holds that the  $\mathcal{DSI}$  algorithm can compute  $\gamma = \frac{\mu'_2 - \mu_2}{c - c'}$  and thus, using the internal state of the signer, produce the rest of the proof.

LEMMA 3.11. If the discrete logarithm problem is  $(t, \varepsilon)$ -hard in the group  $\mathbb{G}$ , then ACL is  $(t', \varepsilon')$ -Exc secure in the AGM + ROM with  $t' \approx t$  and

$$\varepsilon' = \varepsilon + \frac{2Q_{\mathsf{H}_{\mathsf{sh}}} + \kappa + Q_{\mathsf{H}_{\mathsf{reg}}} + \ell}{p}$$

where  $Q_{H_{sh}}$  is the number of queries the adversary makes to  $H_{sh}$ ,  $Q_{H_{reg}}$  is the number if queries the adversary makes to  $H_{reg}$ ,  $\ell$  is the number of signing sessions closed with the user oracles,  $\kappa$  is the number of registered users through the registration oracle.

*Proof Idea.* In order to successfully frame a user, the adversary needs to output a signature along with two showing proofs  $\bar{\pi}_1, \bar{\pi}_2$  as the  $\mathcal{DSI}$  algorithm and provide values  $\gamma$  and rnd that blinds the user's commitment into  $\zeta_1$ . As the signer can interact with itself to issue a signature, thus knowing  $\gamma$  and rnd, we show that it is hard for the signer to provide the partial proofs of opening of the commitment by extracting a discrete logarithm solution from a signer that does so.

**PROOF.** We do a quick game hop to exclude some situations where the reduction cannot win related to RO queries.

#### *Game* $G_1$ . This is the original DSP-game.

*Game* G<sub>2</sub>. This game aborts whenever the challenger queries the hash oracles H<sub>reg</sub> and H<sub>sh</sub> during computation of a proof  $\pi$  or  $\bar{\pi}$ , and the hash oracle has been queried on the same value before. As the challenger's queries to its own oracle contain first flows of the proofs  $\pi$ ,  $\bar{\pi}$ , the probability for each first flow to have been queried before is at most  $\frac{Q_{\text{Hreg}}+\kappa}{p}$  for H<sub>reg</sub> and  $\frac{Q_{\text{Hsh}}+\ell}{p}$  for H<sub>sh</sub>.

*Game* **G**<sub>3</sub>. This game introduces an additional abort condition related to the representations submitted by the adversary to the oracle. When the adversary outputs a proof of double spending  $(C, ID, \pi, \text{rnd}, \gamma, m, \sigma, \pi_1, \pi_2)$ , at least one of  $\pi_1$  and  $\pi_2$  was not generated by the showing oracle. Let  $\bar{\pi} = (\pi_{sdl}, \pi_{op})$  be the showing proof that was not an output of the showing oracle. We consider the  $\pi_{op} = (R_{op}, c, \vec{S}_{op})$  component.

We denote by  $G_{[R]}$  the exponent of a group element *G* in the representation of *R*, where *G* can be any group element that was output by the challenger to the adversary through one of the oracles. The game aborts if

$$c = \sum_{j=1}^{\kappa} C_{j[R]} + \pi.c \cdot \pi.R_{j[R]} + \sum_{j=1}^{\ell} \gamma_j \cdot \zeta_{1,j[R]} + \sum_{j=1}^{\ell} \pi_{op}.c_j \cdot R_{j[R]},$$

where  $\pi_{op}.c_j$  is the value *c* computed as part of the showing,  $\gamma_j$  is the  $\gamma$  value chosen in the  $j_{th}$  signing session,  $\kappa$  is the number of registered users, and  $C_j$  is the commitment of the  $j_{th}$  registered user,  $\pi.R_j$  is the first flow of the proof of (partial) opening of the registration commitment of the  $j_{th}$  registered user, and *c* is the

hash output used in  $\pi_{op}$ . As the representations of  $\gamma_j$  and  $\pi.c$  are already fixed by the time the adversary receives the hash output, the probability of the above event happening is at most  $\frac{Q_{H_{sh}}}{p}$ .

We describe a reduction that breaks the discrete logarithm problem if the adversary wins  $G_3$ .

- The reduction simulates the G<sub>3</sub> game as follows to the adversary:
  - **Setup.** The reduction takes as input a discrete logarithm challenge g, X. It invokes the adversary  $\mathcal{A}$  with the oracles described below. Keep a list of all group elements submitted by the adversary to the oracles along with their algebraic representations.
  - **Online Phase.** The reduction simulates the oracles as follows to the adversary:
    - **Hash oracles**  $H_{pp}$ ,  $H_{reg}$ ,  $H_1$ ,  $H_3$ ,  $H_{sh}$  lazy sampling from the appropriate image spaces, in case of group element image spaces, sample the discrete logarithm to base *g* of the group element from  $\mathbb{Z}_p$  and keep the list of discrete logarithms along with the inputs and group element oracle outputs.
    - **Registration queries** Takes as input the vector to be registered  $\vec{L}$ . Sample  $r_{com'} \leftarrow_{\$} \mathbb{Z}_p$ . Compute the commitment  $C := h_0^{r_{com'}} X \cdot \prod_{i=1}^n h_i^{L_i}$  which implicitly sets  $r_{com} := r_{com'} \cdot \text{dlog} X$ . Simulate the proof  $\pi$  by programming the RO H<sub>reg</sub>.
    - **User queries** participate in the signing protocol like an honest user would and output the resulting signatures to the signer.
    - **Showing queries** On input of a signature previously output by the user oracle and an opening set  $I_{op}$ , produce an opening  $\bar{\pi}$  to this opening index set by simulating the proof by programming the random oracle  $H_{sh}$ .
  - **Output Determination.** The reduction retrieves the algebraic representation of *R* that the adversary submitted when querying the random oracle  $H_{sh}$  to generate  $\bar{\pi}$ . It computes a so-called *reduced representation*<sup>9</sup> of *R* to the base  $\vec{I} = (g, X)$  as follows: We write

$$R = X^{X[R]}\overline{i} \cdot q^{g[R]}\overline{i},$$

where

$$X_{[R]\vec{I}} := \sum_{j=1}^{\kappa} C_{j[R]} + \pi.c \cdot \pi.R_{j[R]} + \sum_{j=1}^{\ell} \gamma_{j} \cdot \zeta_{1,j[R]} + \sum_{j=1}^{\ell} \pi_{op}.c_{j} \cdot R_{j[R]}$$
  
and

$$g_{[X]\vec{I}} \coloneqq \operatorname{dlog}_g(R \cdot X^{-X_{[R]\vec{I}}})$$

The reduction can compute  $g_{[X]\vec{l}}$  using the discrete logarithms of all group elements in its output to the base g, as any X components have been removed from this equation. If  $c = X_{[R]\vec{l}}$ , the reduction aborts as this corresponds to the abort condition introduced in G<sub>3</sub>. Otherwise, it computes

$$\mathrm{dlog}_{g}X = \frac{g_{[R]\vec{I}} - \left(\sum_{i=0}^{n} s_i + s_{\Gamma} + c \cdot \left(r_{com'} + \sum_{i=1}^{n} L_i + \mathrm{rnd}\right)\right)}{c - X_{[R]\vec{I}}}.$$

<sup>&</sup>lt;sup>9</sup>This strategy has been used before in [35, 37].

The reduction outputs a valid DLP solution whenever the adversary wins  $G_3$ . Thus, putting together the probabilities from the game hops yields

$$\operatorname{Adv}_{\mathbb{G}}^{\mathsf{DLP}}(\mathcal{R}) = \operatorname{Adv}_{\mathsf{ACL}}^{\mathsf{Exc}}(\mathcal{A}) - \frac{Q_{\mathsf{H}_{\mathsf{sh}}}}{p} - \frac{Q_{\mathsf{H}_{\mathsf{reg}}} + \kappa}{p} - \frac{Q_{\mathsf{H}_{\mathsf{sh}}} + \ell}{p}$$

which yields the claim.

#### ACKNOWLEDGMENTS

Foteini Baldimtsi was funded through NSF #2143287. Julia Kastner is supported by the Dutch Research Agenda (NWA) project HAPKIDO (Project No. NWA.1215.18.002), which is financed by the Dutch Research Council (NWO). Omar Renawi and Julian Loss were funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 507237585, and by the European Union, ERC-2023-STG, Project ID: 101116713. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

#### REFERENCES

- Masayuki Abe. 2001. A Secure Three-Move Blind Signature Scheme for Polynomially Many Signatures. In *EUROCRYPT 2001 (LNCS, Vol. 2045)*, Birgit Pfitzmann (Ed.). Springer, Berlin, Heidelberg, 136–151. https://doi.org/10.1007/3-540-44987-6.9
- [2] Foteini Baldimtsi, Jan Camenisch, Maria Dubovitskaya, Anna Lysyanskaya, Leonid Reyzin, Kai Samelin, and Sophia Yakoubov. 2017. Accumulators with Applications to Anonymity-Preserving Revocation. In 2017 IEEE European Symposium on Security and Privacy, EuroS&P 2017, Paris, France, April 26-28, 2017. IEEE, 301-315. https://doi.org/10.1109/EUROSP.2017.13
- [3] Foteini Baldimtsi and Anna Lysyanskaya. 2013. Anonymous credentials light. In ACM CCS 2013, Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung (Eds.). ACM Press, 1087–1098. https://doi.org/10.1145/2508859.2516687
- [4] Foteini Baldimtsi and Anna Lysyanskaya. 2013. On the Security of One-Witness Blind Signature Schemes. In ASIACRYPT 2013, Part II (LNCS, Vol. 8270), Kazue Sako and Palash Sarkar (Eds.). Springer, Berlin, Heidelberg, 82–99. https://doi. org/10.1007/978-3-642-42045-0\_5
- [5] Mira Belenkiy, Jan Camenisch, Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Hovav Shacham. 2009. Randomizable Proofs and Delegatable Anonymous Credentials. In CRYPTO 2009 (LNCS, Vol. 5677), Shai Halevi (Ed.). Springer, Berlin, Heidelberg, 108–125. https://doi.org/10.1007/978-3-642-03356-8\_7
- [6] Fabrice Benhamouda, Tancrède Lepoint, Michele Orrù, and Mariana Raykova. 2022. Publicly verifiable anonymous tokens with private metadata bit. Cryptology ePrint Archive, Report 2022/004. https://eprint.iacr.org/2022/004
- [7] Patrik Bichsel, Jan Camenisch, Thomas Groß, and Victor Shoup. 2009. Anonymous credentials on a standard java card. In ACM CCS 2009, Ehab Al-Shaer, Somesh Jha, and Angelos D. Keromytis (Eds.). ACM Press, 600–610. https: //doi.org/10.1145/1653662.1653734
- [8] Johannes Blömer, Jan Bobolz, Denis Diemert, and Fabian Eidens. 2019. Updatable Anonymous Credentials and Applications to Incentive Systems. In ACM CCS 2019, Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz (Eds.). ACM Press, 1671–1685. https://doi.org/10.1145/3319535.3354223
- [9] Dan Boneh, Xavier Boyen, and Hovav Shacham. 2004. Short Group Signatures. In CRYPTO 2004 (LNCS, Vol. 3152), Matthew Franklin (Ed.). Springer, Berlin, Heidelberg, 41–55. https://doi.org/10.1007/978-3-540-28628-8\_3
- [10] Jonathan Bootle, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Alessandro Sorniotti. 2023. A Framework for Practical Anonymous Credentials from Lattices. In CRYPTO 2023, Part II (LNCS, Vol. 14082), Helena Handschuh and Anna Lysyanskaya (Eds.). Springer, Cham, 384–417. https://doi.org/10.1007/978-3-031-38545-2\_13
- [11] Stefan Brands. 1994. Untraceable Off-line Cash in Wallets with Observers (Extended Abstract). In CRYPTO'93 (LNCS, Vol. 773), Douglas R. Stinson (Ed.). Springer, Berlin, Heidelberg, 302–318. https://doi.org/10.1007/3-540-48329-2\_26
- [12] Joakim Brorsson, Bernardo David, Lorenzo Gentile, Elena Pagnin, and Paul Stankovski Wagner. 2023. PAPR: Publicly Auditable Privacy Revocation for Anonymous Credentials. In CT-RSA 2023 (LNCS, Vol. 13871), Mike Rosulek (Ed.). Springer, Cham, 163–190. https://doi.org/10.1007/978-3-031-30872-7\_7
- [13] Jan Camenisch, Susan Hohenberger, Markulf Kohlweiss, Anna Lysyanskaya, and Mira Meyerovich. 2006. How to win the clonewars: Efficient periodic n-times anonymous authentication. In ACM CCS 2006, Ari Juels, Rebecca N. Wright, and

Sabrina De Capitani di Vimercati (Eds.). ACM Press, 201–210. https://doi.org/10.1145/1180405.1180431

- [14] Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. 2005. Compact E-Cash. In EUROCRYPT 2005 (LNCS, Vol. 3494), Ronald Cramer (Ed.). Springer, Berlin, Heidelberg, 302–321. https://doi.org/10.1007/11426639\_18
- [15] Jan Camenisch, Markulf Kohlweiss, and Claudio Soriente. 2010. Solving Revocation with Efficient Update of Anonymous Credentials. In SCN 10 (LNCS, Vol. 6280), Juan A. Garay and Roberto De Prisco (Eds.). Springer, Berlin, Heidelberg, 454–471. https://doi.org/10.1007/978-3-642-15317-4\_28
- [16] Jan Camenisch and Anna Lysyanskaya. 2001. An Efficient System for Nontransferable Anonymous Credentials with Optional Anonymity Revocation. In EUROCRYPT 2001 (LNCS, Vol. 2045), Birgit Pfitzmann (Ed.). Springer, Berlin, Heidelberg, 93–118. https://doi.org/10.1007/3-540-44987-6\_7
- [17] Jan Camenisch and Anna Lysyanskaya. 2004. Signature Schemes and Anonymous Credentials from Bilinear Maps. In *CRYPTO 2004 (LNCS, Vol. 3152)*, Matthew Franklin (Ed.). Springer, Berlin, Heidelberg, 56–72. https://doi.org/10.1007/978-3-540-28628-8\_4
- [18] Melissa Chase, F. Betül Durak, and Serge Vaudenay. 2023. Anonymous Tokens with Stronger Metadata Bit Hiding from Algebraic MACs. In CRYPTO 2023, Part II (LNCS, Vol. 14082), Helena Handschuh and Anna Lysyanskaya (Eds.). Springer, Cham, 418–449. https://doi.org/10.1007/978-3-031-38545-2\_14
- [19] Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Sarah Meiklejohn. 2014. Malleable Signatures: New Definitions and Delegatable Anonymous Credentials. In CSF 2014 Computer Security Foundations Symposium, Anupam Datta and Cedric Fournet (Eds.). IEEE Computer Society Press, 199–213. https://doi.org/10.1109/CSF.2014.22
- [20] Melissa Chase, Trevor Perrin, and Greg Zaverucha. 2020. The Signal Private Group System and Anonymous Credentials Supporting Efficient Verifiable Encryption. In ACM CCS 2020, Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna (Eds.). ACM Press, 1445–1459. https://doi.org/10.1145/3372297.3417887
- [21] David Chaum. 1982. Blind Signatures for Untraceable Payments. In CRYPTO'82, David Chaum, Ronald L. Rivest, and Alan T. Sherman (Eds.). Plenum Press, New York, USA, 199–203. https://doi.org/10.1007/978-1-4757-0602-4\_18
- [22] David Chaum. 1983. Blind Signature System. In CRYPTO'83, David Chaum (Ed.). Plenum Press, New York, USA, 153. https://doi.org/10.1007/978-1-4684-4730-9\_14
- [23] Ronald Cramer. 1997. Modular Design of Secure yet Practical Cryptographic Protocols. Ph. D. Dissertation.
- [24] Elizabeth C. Crites and Anna Lysyanskaya. 2019. Delegatable Anonymous Credentials from Mercurial Signatures. In CT-RSA 2019 (LNCS, Vol. 11405), Mitsuru Matsui (Ed.). Springer, Cham, 535–555. https://doi.org/10.1007/978-3-030-12612-4\_27
- [25] Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. 2018. Privacy Pass: Bypassing Internet Challenges Anonymously. *PoPETs* 2018, 3 (July 2018), 164–180. https://doi.org/10.1515/popets-2018-0026
- [26] Jack Doerner, Yashvanth Kondi, Eysa Lee, abhi shelat, and LaKyah Tyner. 2023. Threshold BBS+ Signatures for Distributed Anonymous Credential Issuance. In 2023 IEEE Symposium on Security and Privacy. IEEE Computer Society Press, 773–789. https://doi.org/10.1109/SP46215.2023.10179470
- [27] Amos Fiat and Adi Shamir. 1987. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *CRYPTO'86 (LNCS, Vol. 263)*, Andrew M. Odlyzko (Ed.). Springer, Berlin, Heidelberg, 186–194. https://doi.org/10.1007/3-540-47721-7\_12
- [28] Georg Fuchsbauer, Eike Kiltz, and Julian Loss. 2018. The Algebraic Group Model and its Applications. In CRYPTO 2018, Part II (LNCS, Vol. 10992), Hovav Shacham and Alexandra Boldyreva (Eds.). Springer, Cham, 33–62. https://doi.org/10.1007/ 978-3-319-96881-0\_2
- [29] Georg Fuchsbauer, David Pointcheval, and Damien Vergnaud. 2009. Transferable Constant-Size Fair E-Cash. In Cryptology and Network Security, 8th International Conference, CANS 2009, Kanazawa, Japan, December 12-14, 2009. Proceedings (Lecture Notes in Computer Science, Vol. 5888), Juan A. Garay, Atsuko Miyaji, and Akira Otsuka (Eds.). Springer, 226–247. https://doi.org/10.1007/978-3-642-10433-6\_15
- [30] Christina Garman, Matthew Green, and Ian Miers. 2014. Decentralized Anonymous Credentials. In NDSS 2014. The Internet Society. https://doi.org/10.14722/ ndss.2014.23253
- [31] Lucjan Hanzlik and Daniel Slamanig. 2021. With a Little Help from My Friends: Constructing Practical Anonymous Credentials. In ACM CCS 2021, Giovanni Vigna and Elaine Shi (Eds.). ACM Press, 2004–2023. https://doi.org/10.1145/ 3460120.3484582
- [32] Gesine Hinterwälder, Christian T. Zenger, Foteini Baldimtsi, Anna Lysyanskaya, Christof Paar, and Wayne P. Burleson. 2013. Efficient E-Cash in Practice: NFC-Based Payments for Public Transportation Systems. In PETS 2013 (LNCS, Vol. 7981), Emiliano De Cristofaro and Matthew K. Wright (Eds.). Springer, Berlin, Heidelberg, 40–59. https://doi.org/10.1007/978-3-642-39077-7\_3

- [34] Ioanna Karantaidou, Omar Renawi, Foteini Baldimtsi, Nikolaos Kamarinakis, Jonathan Katz, and Julian Loss. 2024. Blind Multisignatures for Anonymous Tokens with Decentralized Issuance. In Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, CCS 2024, Salt Lake City, UT, USA, October 14-18, 2024, Bo Luo, Xiaojing Liao, Jun Xu, Engin Kirda, and David Lie (Eds.). ACM, 1508–1522. https://doi.org/10.1145/3658644.3690364
- [35] Julia Kastner, Julian Loss, and Omar Renawi. 2023. Concurrent Security of Anonymous Credentials Light, Revisited. In Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (Copenhagen, Denmark) (CCS '23). Association for Computing Machinery, New York, NY, USA, 45–59. https://doi.org/10.1145/3576915.3623184
- [36] Julia Kastner, Julian Loss, and Omar Renawi. 2023. Concurrent Security of Anonymous Credentials Light, Revisited. Cryptology ePrint Archive, Report 2023/707. https://eprint.iacr.org/2023/707
- [37] Julia Kastner, Julian Loss, and Jiayu Xu. 2022. On Pairing-Free Blind Signature Schemes in the Algebraic Group Model. In *PKC 2022, Part II (LNCS, Vol. 13178)*, Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe (Eds.). Springer, Cham, 468–497. https://doi.org/10.1007/978-3-030-97131-1\_16
- [38] Ben Kreuter, Tancrède Lepoint, Michele Orrù, and Mariana Raykova. 2020. Anonymous Tokens with Private Metadata Bit. In *CRYPTO 2020, Part I (LNCS, Vol. 12170)*, Daniele Micciancio and Thomas Ristenpart (Eds.). Springer, Cham, 308–336. https://doi.org/10.1007/978-3-030-56784-2\_11
- [39] Tatsuaki Okamoto and Kazuo Ohta. 1992. Universal Electronic Cash. In CRYPTO'91 (LNCS, Vol. 576), Joan Feigenbaum (Ed.). Springer, Berlin, Heidelberg, 324–337. https://doi.org/10.1007/3-540-46766-1\_27
- [40] Michele Orrù, Stefano Tessaro, Greg Zaverucha, and Chenzhi Zhu. 2024. Oblivious Issuance of Proofs. In CRYPTO 2024, Part IX (LNCS, Vol. 14928), Leonid Reyzin and Douglas Stebila (Eds.). Springer, Cham, 254–287. https://doi.org/10.1007/978-3-031-68400-5\_8
- [41] Christian Paquin and Greg Zaverucha. 2013. U-Prove Technology Overview V1.1. In Microsoft Technical Report, https://www.microsoft.com/enus/research/publication/uprove-cryptographic-specification-v1-1-revision-3/.
- [42] Torben P. Pedersen. 1992. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In CRYPTO'91 (LNCS, Vol. 576), Joan Feigenbaum (Ed.). Springer, Berlin, Heidelberg, 129–140. https://doi.org/10.1007/3-540-46766-1\_9
- [43] Alfredo Rial and Ania M. Piotrowska. 2023. Compact and Divisible E-Cash with Threshold Issuance. Proc. Priv. Enhancing Technol. 2023, 4 (2023), 381–415. https://doi.org/10.56553/POPETS-2023-0116
- [44] Olivier Sanders. 2020. Efficient Redactable Signature and Application to Anonymous Credentials. In PKC 2020, Part II (LNCS, Vol. 12111), Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas (Eds.). Springer, Cham, 628–656. https://doi.org/10.1007/978-3-030-45388-6\_22
- [45] Tjerand Silde and Martin Strand. 2022. Anonymous Tokens with Public Metadata and Applications to Private Contact Tracing. In FC 2022 (LNCS, Vol. 13411), Ittay Eyal and Juan A. Garay (Eds.). Springer, Cham, 179–199. https://doi.org/10.1007/ 978-3-031-18283-9\_9
- [46] Alberto Sonnino, Mustafa Al-Bassam, Shehar Bano, Sarah Meiklejohn, and George Danezis. 2019. Coconut: Threshold Issuance Selective Disclosure Credentials with Applications to Distributed Ledgers. In NDSS 2019. The Internet Society. https://doi.org/10.14722/ndss.2019.23272
- [47] Alin Tomescu, Adithya Bhat, Benny Applebaum, Ittai Abraham, Guy Gueta, Benny Pinkas, and Avishay Yanai. 2022. UTT: Decentralized Ecash with Accountable Privacy. *IACR Cryptol. ePrint Arch.* (2022), 452. https://eprint.iacr.org/ 2022/452

# A ADDITIONAL PRELIMINARIES

# A.1 Proofs for Pedersen Commitments

LEMMA 2.8 (PEDERSEN COMMITMENTS ARE PERFECTLY HIDING). The commitment scheme GPC from definition 2.6 is perfectly hiding.

PROOF. Information-theoretically, for each vector of commitment parameters  $\vec{h} \in \mathbb{G}^{n+1}$ , each commitment  $C \in \mathbb{G}$  and each value vector  $v \in \mathbb{Z}_p^n$ , there exists exactly one opening  $r = \text{dlog}_{h_0} \frac{C}{\prod_{i=1}^n h_i^{v_i}}$ . The commitment thus perfectly hides the vector v and even an unbounded adversary cannot do better than guessing.  $\Box$ 

LEMMA 2.9 (PEDERSEN COMMITMENTS ARE BINDING UNDER DL). If the DLP is  $(t, \varepsilon)$ -hard in the group G, GPC from definition 2.6 is  $(t', \varepsilon')$ -binding with  $t' \approx t$  and  $\varepsilon' \leq n \cdot \varepsilon$ . PROOF. We provide a reduction  $\mathcal{B}$  against DLP that uses a binding adversary  $\mathcal{A}$  to break DLP. The reduction takes as input the discrete logarithm challenge g, Y. It samples  $j \leftarrow_{\$} [n]$  and sets  $h_j := Y$ . For each  $i \neq j \in [n]_0$  it samples  $a_i \leftarrow_{\$} \mathbb{Z}_p$  and sets  $h_i := g^{a_i}$ . It outputs the commitment parameters  $\operatorname{pp}_{\operatorname{GPC}} = (h_0, \ldots, h_n)$  to the adversary  $\mathcal{A}$ . When the adversary outputs a commitment C and two vectors  $\vec{v}_1, \vec{v}_2$  along with openings  $r_1, r_2$ , the reduction aborts if  $v_{1,j} = v_{2,j}$ . As  $\vec{v}_1 \neq \vec{v}_2$ , there must be at least one index  $i^*$  where  $v_{1,i^*} \neq v_{2,i^*}$  and  $j = i^*$  happens with probability  $\frac{1}{n}$ . Thus, the reduction aborts with probability at most  $1 - \frac{1}{n}$ . If the reduction did not abort in the previous step, it computes dlog  $Y = \frac{r_1 \cdot a_0 + \sum_{i \neq j} a_i \cdot v_{1,i} - (r_2 \cdot a_0 + \sum_{i \neq j} a_i \cdot v_{2,i})}{v_{2,j} - v_{1,j}}$  and outputs the solution.

## A.2 Sigma-Protocols and Proofs

We recall definition of  $\Sigma$ -protocols and some of their security properties, see also for example [23]

Definition A.1 ( $\Sigma$ -Protocol). A  $\Sigma$ -Protocol for a relation  $\mathcal{R}$  consists of the following algorithms  $\mathcal{P} = (\mathcal{P}_1, \mathcal{P}_2), \mathcal{V} = (\mathcal{V}_1, \mathcal{V}_2).$ 

- $\mathcal{P}_1(x, w)$  Outputs a message *R* and saves an internal state  $s_{\mathcal{P}}$  $\mathcal{V}_1(x, R)$  Takes as input the statement and the prover's first message and outputs a challenge  $c \leftarrow_{\mathbb{S}} C$ .
- $\mathcal{O}$  (*u*, *u*, *u*, *u*)  $\mathcal{O}$ -tractice *u* chanceling *u*  $\leftarrow$  §
- $\mathcal{P}_2(x, w, s_{\mathcal{P}}, c)$  Outputs a response S.
- $\mathcal{V}_2(x, R, c, S)$  Outputs a bit *b* indicating acceptance or rejection.

Definition A.2 ( $\Sigma$ -Protocol Correctness). A  $\Sigma$ -Protocol is correct if for all  $(x, w) \in \mathcal{R}$ , all  $(R, s_{\mathcal{P}}) \leftarrow_{\$} \mathcal{P}_1(x, w)$ , all  $c \in C$ , and all  $S \leftarrow_{\$} \mathcal{P}_2(x, w, s_{\mathcal{P}}, c)$ , it holds that  $\mathcal{V}_2(x, R, c, S) = 1$ 

Definition A.3. (HVZK) A  $\Sigma$ -Protocol is honest-verifier zero-knowledge (HVZK) if there exists a simulator S such that for all  $x \in \mathcal{L}$  and all  $c \in C$ , the distributions of  $(R, c, S) \leftarrow_{\$} S(x, c)$  are indistinguishable from  $R' \leftarrow_{\$} \mathcal{P}_1(x, w), c, S \leftarrow_{\$} \mathcal{P}_2(x, w, s_{\mathcal{P}}, c)$  conditioned on c being used as the challenge.

Definition A.4 (k-Special-Soundness). A  $\Sigma$ -Protocol is k-special sound if there exists a deterministic extractor  $\mathcal{E}$  such that given k valid transcripts  $\{(R, c_i, S_i)\}_{i \in [k]}$  for statement x with pairwise distinct challenges  $c_i$ , outputs a witness w such that  $(x, w) \in \mathcal{R}$ .

Definition A.5 ( $\varepsilon$ -Soundness). A  $\Sigma$ -Protocol is  $\varepsilon$ -sound if for any  $x \notin \mathcal{L}$ , any adversary  $\mathcal{P}^*$  it holds that the probability that  $\mathcal{V}_2(x, R, c, S) = 1$  is at most  $\varepsilon$ . Where  $(R, s_{\mathcal{P}}) \leftarrow_{\$} \mathcal{P}_1^*(x), c \leftarrow_{\$} \mathcal{V}_1(x, R)$ , and  $S \leftarrow_{\$} \mathcal{P}_2(x, w, s_{\mathcal{P}}, c)$  and the probability is taken over the random coins of the  $\mathcal{P}^*$  and  $\mathcal{V}_1(x, R)$ .

Definition A.6 (Fiat Shamir Transform [27]). The Fiat-Shamir transform of a  $\Sigma$ -protocol  $\Sigma = (\mathcal{P} = (\mathcal{P}_1, \mathcal{P}_2), \mathcal{V} = (\mathcal{V}_1, \mathcal{V}_2))$  with challenge space *C* and a hash function H:  $\{0, 1\}^* \to C$  and some auxiliary information that might be used in surrounding protocols is described below.

- $\mathcal{P}(x, w)$  Compute  $(R, s_{\mathcal{P}}) \leftarrow_{\$} \mathcal{P}_1(x, w)$ . Compute c := H(x, R, aux) for potential auxiliary information *aux*. Compute  $S \leftarrow_{\$} \mathcal{P}_2(x, w, s_{\mathcal{P}}, c)$ . Output  $\pi = (R, S)$
- $\mathcal{V}(x, \pi = (R, S), aux)$  Compute c = H(x, R, aux), output  $\mathcal{V}_2(x, R, c, S)$ .

Secure Showing of Partial Attributes

LEMMA A.7. Let  $\Sigma_1$  be a  $\Sigma$ -protocol for language  $\mathcal{L}_1$  with relation  $\mathcal{R}_1$  and  $\Sigma_2$  be a  $\Sigma$ -protocol for language  $\mathcal{L}_2$  with relation  $\mathcal{R}_2$  where  $\Sigma_1$  and  $\Sigma_2$  have the same challenge space C and both fulfill k-Special-Soundness and HVZK as above. Then, the following protocol is a  $\Sigma$ -protocol for the language  $\mathcal{L}_1 \times \mathcal{L}_2$  with relation  $\mathcal{R} = \{((x_1, x_2, ), (w_1, w_2)): (x_1, w_1) \in \mathcal{R}_1, (x_2, w_2) \in \mathcal{R}_2\}$  which fulfills k-Special-Soundness and HVZK.

- $\mathcal{P}_1 \text{ compute } (R_1, s_1) \leftarrow_{\$} \Sigma_1.\mathcal{P}_1(x_1, w_1), (R_2, s_2) \leftarrow_{\$} \Sigma_2.\mathcal{P}_2$
- $\mathcal{V}_1 \text{ sample } c \leftarrow_{\$} C$
- $\mathcal{P}_{2} \quad compute \ S_{1} \leftarrow_{\$} \Sigma_{1}.\mathcal{P}_{2}(x_{1}, w_{1}, s_{1}, c), S_{2} \leftarrow_{\$} \Sigma_{2}.\mathcal{P}_{2}(x_{2}, w_{2}, s_{2}, c)$
- $V_2(x_1, x_2, R_1, R_2, c, S_1, S_2)$  runs  $\Sigma_1 . V_2(x_1, R_1, c, S_1)$  and  $\Sigma_2 . V_2(x_2, R_2, c, S_2)$  and output 1 iff both output 1.

PROOF. For HVZK, given  $c, x_1, x_2$  run  $(R_1, S_1) \leftarrow S_1(x_1, c)$  and  $(R_2, S_2) \leftarrow S_2(x_2, c)$  where  $S_1$  and  $S_2$  are the HVZK simulators for  $\Sigma_1$  and  $\Sigma_2$ , respectively.

For *k*-special soundness, given *k* transcripts with the same first flow  $R_1, R_2$  and *k* different challenges  $c_i$  and corresponding responses  $S_{1,i}, S_{2,i}$ , run  $\mathcal{E}_1$  on  $\{(R_1, c_i, S_{1,i}) : i \in [k]\}$  and  $\mathcal{E}_2$  on  $\{(R_2, c_i, S_{2,i}) : i \in [k]\}$  to obtain the witnesses  $w_1$  and  $w_2$  respectively.

## A.3 Identification Scheme from ACL (ID<sub>ACL</sub>)

In the following, we describe the identification scheme  $ID_{ACL}$ , which was proposed by [35]. The scheme is defined as tuple ( $\mathcal{G}, \mathcal{P}, \mathcal{V}$ ), where

- $\mathcal{G}(\text{pp})$ : Parse pp to obtain  $\mathbb{G}, p, g$ , sample  $h \leftarrow_{\$} \mathbb{G}$  and  $x, v_{0,1}, v_{0,2} \leftarrow_{\$} \mathbb{Z}_p$ , compute  $y = g^x, z := g^{v_{0,1}} \cdot h^{v_{0,2}}$ , set sk = x and pk = (h, z, y), and output (pk, sk).
- The prover algorithms  $\mathcal{P}_{=}(\mathcal{P}_{1}, \mathcal{P}_{2})$  are defined as follows: -  $\mathcal{P}_{1}(sk)$ : Sample  $d, s_{1}, s_{2}, u, \operatorname{rnd} \leftarrow_{\$} \mathbb{Z}_{p}$ , compute  $z_{1} := g^{\operatorname{rnd}}, z_{2} := z/z_{1}, a := g^{u}, b_{1} := g^{s_{1}} \cdot z_{1}^{d}, b_{2} := h^{s_{2}} \cdot z_{2}^{d},$ set  $R := (a, b_{1}, b_{2}, \operatorname{rnd}), st_{p} := (d, s_{1}, s_{2}, u)$  and output  $(R, st_{p})$ .
  - $\mathcal{P}_2(\mathsf{sk}, \mathsf{st}_p, \mathsf{R}, \varepsilon)$ : Parse  $\mathsf{st}_p$  as  $(d, \mathsf{s}_1, \mathsf{s}_2, u)$ , compute  $c := \varepsilon d, r := u c \cdot \mathsf{sk}$ , set and output  $S := (c, d, r, \mathsf{s}_1, \mathsf{s}_2)$ .
- The verifier algorithms V=(V<sub>1</sub>, V<sub>2</sub>) are defined as follows:
   V<sub>1</sub>(pk, R): Parse pk, and R = (a, b<sub>1</sub>, b<sub>2</sub>, rnd). Output ⊥ if rnd = 0. Sample ε ←<sub>\$</sub> ℤ<sub>p</sub>, set st<sub>v</sub> := (a, b<sub>1</sub>, b<sub>2</sub>, z<sub>1</sub> := g<sup>rnd</sup>, z<sub>2</sub> := z/z<sub>1</sub>), and output (ε, st<sub>v</sub>).
  - $\mathcal{V}_2(\mathsf{pk}, st_v, R, \varepsilon, S)$ : Parse *S* and  $st_v$ , and output 1 if the following condition holds:  $\varepsilon = c + d \wedge a = g^r \cdot y^c \wedge b_1 = g^{s_1} \cdot z_1^d \wedge b_2 = h^{s_2} \cdot z_2^d$ . Otherwise, output 0.

# **B** PARTIALLY OPENING BLINDED PEDERSEN COMMITMENTS

#### **B.1** Proof of Same Discrete Logarithm

Definition B.1 (Sigma Protocol for Proof of Same Discrete Logarithm). We describe how a prover  $\mathcal{P}$  can prove in zero knowledge that the elements of a vector of group elements  $\vec{h} = (h_1, \ldots, h_n)$  have the same discrete logarithms to the bases of another vector of group elements  $\vec{h'} = (h'_1, \ldots, h'_n)$ . That is  $dlog_{h_1}h'_1 = \ldots = dlog_{h_n}h'_n$ .

- $\mathcal{P}_1(x = (\vec{h}, \vec{h}'), w = \gamma)$  The prover takes as input two vectors of the same length and a witness  $\gamma$  which is the discrete logarithm relation between them. It samples  $r_{sdl} \leftarrow_{\$} \mathbb{Z}_p$ and sets  $\vec{h}_{sdl} = \vec{h}^{r_{sdl}}$ , i.e. the component wise exponentiation of  $\vec{h}$  with  $r_{sdl}$ . It outputs the commitment vector  $\vec{h_{sdl}}$ to the verifier.
- $\mathcal{V}_1(x = (h, h'), h_{sdl})$  The verifier takes as input the statement and the prover's first message. It samples a random value  $c \leftarrow_{\$} \mathbb{Z}_p$  and outputs it to  $\mathcal{P}$ .
- $\mathcal{P}_2(x = (\vec{h}, \vec{h}'), w = \gamma, r_{sdl}, c)$  The prover takes as input the statement  $\vec{h}, \vec{h}'$ , the witness  $\gamma$ , its internal state  $r_{sdl}$  and the verifier's challenge *c*. It computes  $s_{sdl} = r_{sdl} c \cdot \gamma$ .
- $\mathcal{V}_2(x = (\vec{h}, \vec{h}'), \vec{h}_{sdl}, c, s_{sdl})$  The verifier checks that  $\vec{h}_{sdl} = \vec{h}^{s_{sdl}} \cdot \vec{h}'^c$  where exponentiation is taken component wise. It returns 1 if the above equation holds and 0 otherwise.

LEMMA B.2 (SOUNDNESS OF DEFINITION B.1). The protocol from definition B.1 is  $\frac{1}{p}$ -sound.

PROOF. Let  $\vec{h}, \vec{h}'$  be two vectors such that there exists indices i, jsuch that  $\gamma_i = \text{dlog}_{h_i} h'_i \neq \text{dlog}_{h_j} h'_j = \gamma_j$ . Let  $\vec{h}_{sdl}^*$  be an adversarial prover's first message. Let  $r_i = \text{dlog}_{h_i} h^*_{sdl,i}$  and  $r_j = \text{dlog}_{h_j} h^*_{sdl,j}$ . Then, there exists at most one  $c^* \in \mathbb{Z}_p$  such that there exists a  $s_{sdl}$ such that  $r_i = s_{sdl} + \gamma_i \cdot c^*$  and  $r_j = s_{sdl} + \gamma_j \cdot c^*$ . The probability that the hones verifier samples  $c = c^*$  is  $\frac{1}{p}$ .

Lemma B.3 (Special Knowledge Soundness of definition B.1). *The protocol from definition B.1 is 2-special sound.* 

**PROOF.** Given two transcripts  $\vec{h}_{sdl}$ ,  $c_1$ ,  $s_1$  and  $\vec{h_{sdl}}$ ,  $c_2$ ,  $s_2$  with the same initial message  $\vec{h}_{sdl}$ , the extractor computes  $\gamma = \frac{s_1 - s_2}{c_2 - c_1}$ .

LEMMA B.4 (EXTRACTION IN THE AGM). Let  $\mathcal{P}^*$  be an algebraic adversary acting as the prover in the protocol from definition B.1 that interacts with a verifier. Then there exists an extractor  $\mathcal{E}_{alg}$  that extracts a witness  $\gamma$  with probability  $1 - \frac{1+n}{p}$  from the algebraic adversary.

PROOF. The algebraic prover takes as input the group elements  $\vec{h}$ . It outputs a vector  $\vec{h}'$  along with algebraic representations  $\vec{a}_i$  s.t.  $h'_i = \prod h_i^{a_i}$ . For each proof transcript, the prover outputs a vector  $\vec{h}_{sdl}$  along with algebraic representations  $\{\vec{z}_i\}_{i \in [n]}$  for each element  $h_{sdl,i}$  to the basis  $\vec{h}$ . the verifier then picks a challenge  $c \leftarrow_{\$} \mathbb{Z}_p$  uniformly at random.

The prover responds with *s* such that for all  $i \in [n]$ , it holds that  $h_i^s \cdot h_i'^c = h_{sdl,i}$ .

By lemma B.2, the probability that the prover succeeds in outputting a valid response when  $\vec{h}, \vec{h}'$  are not in the language is  $\frac{1}{p}$ . In the following, we assume that this doesn't happen.

This yields n linear equations of the form

$$\sum_{j=1}^{n} z_{i,j} \cdot \mathrm{dlog} h_j = s \cdot \mathrm{dlog} h_i + c \cdot \sum_{j=1}^{n} a_{i,j} \cdot \mathrm{dlog} h_i \tag{1}$$

where the discrete logarithms are unknown as well as *n* equations of the form  $\gamma \cdot dlogh_i = \sum_{j=1} a_{i,j} \cdot dlogh_i$  where both  $\gamma$  and the discrete logarithms are unknown. If no fixed generator is given for the group, we assume w.l.o.g. that discrete logarithms are taken to the base  $h_1$  (i.e.dlog $h_1 = 1$ ).

W.l.o.g we assume that all  $h_i$  are pairwise distinct and no  $h_i$  is the neutral element. Otherwise, we conflate the equal elements as well as their representations.

We consider the following cases:

- (1)  $\exists i: a_{i,i} \neq 0 \land \forall i' \neq i: a_{i,i'} = 0$ : In this case, the extractor outputs the witness  $\gamma = a_i$ .
- (2) The system of linear equations is linearly dependent. We consider the following subcases:
  - (a) The matrix A consisting of the representation vectors  $\vec{a}_i$  has full rank. We make an information-theoretic argument why the system of equations has to be linearly dependent with probability at most  $\frac{n}{p}$  or with probability 1 pver the choice of *c*.

First, we note that information-theoretically, the choice of *c* already fixes *s*. Also, information-theoretically  $\gamma$  is fixed by the vectors  $\vec{h}$  and  $\vec{h}'$  already, as well as the adversary's implicit choice of  $r_{sdl}$  through  $\vec{h}_{sdl}$ .

The determinant of the matrix *A* with *c*, *s* as formal variables is a polynomial of total degree at most n. As *c* information-theoretically fixes *s*, we can reduce the number of formal variables in the determinant polynomial to 1 by replacing *s* with  $r_{sdl} - c \cdot \gamma$ . This yields a polynomial in one variable *c* of degree at most *n*. If this polynomial is the constant 0 polynomial, the probability of linear dependence is 1, i.e. for any pair *c*, *s* that fulfills the linear relation above, the matrix does not have full rank. In this case the extractor picks a random  $c' \leftarrow_{\$} \mathbb{Z}_p$ . It plugs it into the matrix and computes the determinant of the matrix with s as a variable (but not *c*). It then computes the roots *s*' of the resulting polynomial. This yields at most *n* possible values of s'. The extractor tests (using the verification equation of the  $\Sigma$ -protocol) which one is the correct one. It then solves for the witness  $\gamma = \frac{s-s'}{c'-c}$ . With probability  $1 - \frac{1}{p}$  it holds that  $c' \neq c$  and thus the extraction succeeds.

If it is not the 0 polynomoial, by the Schwartz-Zippel-Lemma, the probability of finding a root of the polynomial when randomly choosing *c* is at most  $\frac{n}{p}$ . In this case, the extractor cannot extract, so this happens with probability  $\frac{n}{p}$ .

(b) The matrix  $\hat{A}$  consisting of the representation vectors  $\vec{a}_i$  does not have full rank. In this case, there exists a linear combination  $\vec{b}$  such that  $\sum_{i=1}^n b_i \cdot \vec{a}_i = 0$ . Due to soundness of the sigma protocol (implying correctness of the statement), this means that with probability  $1 - \frac{1}{p}$  it holds that  $\prod_{i=1}^n h_i^{b_i} = 0$ . We therefore know a way to express one of the  $h_i$  as a linear combination of the others. We remove this  $h_i$  from all systems of equations and replace it with its linear representation of the other  $h_i$ . This yields systems of equations with one variable less. We consecutively apply this transformation until

the matrix A' resulting from these transformations has full rank. If A' has rank 1, we proceed as in Case 1. Otherwise we continue as in Case 2a.

(3) Otherwise: As we excluded Case 1 and Case 2, the system of equations is linearly independent. The extractor can thus compute a the discrete logarithms of all h<sub>i</sub> w.r.t. h<sub>1</sub> by solving the linear equation system. It can use the representation of h'<sub>1</sub> to compute γ = dlog<sub>h1</sub> h'<sub>1</sub> and output γ.

Overall, the failure probability of the extractor can be bounded by  $\frac{n}{p} + \frac{1}{p}$ .

LEMMA B.5 (HVZK OF DEFINITION B.1). The protocol from definition B.1 fullfills special honest verifier zero-knowledge.

PROOF. We give a simulator. On input of a challenge *c* and a statement  $\vec{h}, \vec{h}'$ , the simulator samples  $s_{sdl} \leftarrow_{\$} \mathbb{Z}_p$  and computes  $\vec{h}_{sdl} := \vec{h}^{s_{sdl}} \cdot \vec{h'}^c$  where exponentiation and multiplication is component wise. It is easy to see that for  $\vec{h}, \vec{h'}$  where there exists  $\gamma$  such that  $\vec{h''} = \vec{h'}$  the transcripts generated by the simulator are identically distributed to real transcripts that use *c* as a challenge.  $\Box$ 

#### **B.2** Proofs of Partial Openings

Definition B.6 (Protocol for Partially Opening a Pedersen Commitment). We describe a sigma protocol that allows for partially opening a generalized Pedersen Commitment while proving knowledge of the remainder of the opening.

Let h be the base vector of the Pedersen commitment, and  $\vec{a}$  be the full opening. Let C be the Pedersen commitment obtained by  $C := \prod_{i=1}^{n} h_i^{a_i}$ . For simplicity, we treat the randomness used for committing as part of the full opening. Let I be a set of indices where the commitment is to be opened.

- $\mathcal{P}_1(\vec{h}, \vec{a}, I)$  For all  $j \notin I$ , the prover samples  $r_j \leftarrow_{\$} \mathbb{Z}_p$ . It then computes  $R \coloneqq \prod_{j \notin I} h_j^{r_j}$ . It outputs R and  $A_I \coloneqq \{a_i \colon i \in I\}$  to the verifier.
- $\mathcal{V}_1(\vec{h}, C, A_I, R)$  The verifier takes the prover's first message as input along with the commitment and the parameters  $\vec{h}$ and samples  $c \leftarrow_{\$} \mathbb{Z}_p$ . It sends c to the prover.
- $\mathcal{P}_2(\vec{h}, \vec{a}, I, \{r_j : j \notin I\}, c)$  The prover takes the commitment parameters  $\vec{h}$ , the opening  $\vec{a}$ , the opening indices *i*, and the randomness for his first message  $r_j$ , along with the challenge from the verifier and for all  $j \notin I$  computes  $s_j \coloneqq r_j c \cdot a_j$ . It then outputs  $S = \{s_j : j \notin I\}$  to the verifier.
- $\mathcal{V}_2(\vec{h}, C, A_I, R, c, S)$  The verifier computes  $C' = \frac{C}{\prod_{i \in I} h_i^{a_i}}$  and then checks that  $R = C'^c \cdot \prod_{j \notin I} h_j^{s_j}$ . It outputs 1 if this equality holds and 0 otherwise.

LEMMA B.7 (2-SPECIAL SOUNDNESS OF DEFINITION B.6). The protocol from definition B.6 is 2-special sound.

PROOF. Given two transcripts  $(R, c_1, S_1)$  and  $(R, c_2, S_2)$  with the same initial message, the extractor  $\mathcal{E}$  computes the witness  $a_i$  for each  $i \in I$  as  $a_i = \frac{s_{1,i} - s_{2,i}}{c_2 - c_1}$ .

LEMMA B.8 (HVZK OF DEFINITION B.6). The protocol from definition B.6 is HVZK. PROOF. The simulator works similar to that from lemma B.5.  $\Box$ 

Definition B.9. Let  $\vec{h}$  be the public parameters of a Pedersen commitment and g the generator of the group over which the Pedersen commitment is instantiated. For a Pedersen commitment C,  $C^{\gamma} \cdot g^{r\gamma}$  is a blinded Pedersen commitment with public parameters  $\vec{h}^{\gamma}, g^{\gamma}$ .

LEMMA B.10 (EXTRACTION IN THE AGM). Let  $\mathcal{P}^*$  be an algebraic adversary acting as the prover in the protocol from definition B.6 that interacts with a verifier. Then there exists an extractor  $\mathcal{E}_{alg}$  that extracts a witness  $\{a_i: i \notin I\}$  or a solution to the **BINDING** game of Pedersen commitments with probability  $1 - \frac{2}{p}$  from the algebraic adversary.

**PROOF.** The adversary outputs a commitment *C* along with a representation  $\vec{y}$  such that  $C = \prod_{i=1}^{n} h_i^{y_i}$  In the first round of the protocol, the adversary  $\mathcal{P}^*$  outputs *R* along with a representation  $\vec{z}$  such that  $R = \prod_{i=1}^{n} h_i^{z_i}$  as well as *I*, *A*<sub>*I*</sub>.

We consider the following cases:

- (1) For all  $i \in I$  it holds that  $z_i = 0$ : Then, the extractor simply computes all  $a_i$  where  $i \notin I$   $a_i = \frac{z_i s_i}{c}$ . This works if  $c \neq 0$  which happens with probability  $1 \frac{1}{p}$
- (2) There exists at least one  $i \in I$  such that  $z_i \neq 0$ .
  - (a) For all  $i \in I$  it holds that  $y_i = a_i$ . In this case, it holds that  $C = \prod_{i \in I} h_i^{a_i} \cdot \prod_{i \notin I} h_i^{y_i}$ , and the extractor outputs  $\{y_i : i \notin I\}$  as the witness.
  - (b) This yields an equation of the form ∏<sub>i=1</sub><sup>n</sup> h<sub>i</sub><sup>c·y<sub>i</sub></sup>. ∏<sub>i∉I</sub> h<sub>i</sub><sup>s<sub>i</sub></sup> = ∏<sub>i=1</sub><sup>n</sup> h<sub>i</sub><sup>z<sub>i</sub></sup>. With probability 1 <sup>1</sup>/<sub>p</sub> over the choice of *c* it holds that there exists *i* ∈ *I* such that *z<sub>i</sub>* ≠ *c* · *y<sub>i</sub>*. Thus, the extractor can output a BINDING solution, namely the commitment *R* along with *z* and *z*' where *z*'<sub>i</sub> = *c* · *y<sub>i</sub>* for *i* ∈ *I* and *z*'<sub>i</sub> = *c* · *y<sub>i</sub>* for *i* ∉ *I*.

THEOREM B.11 (OPENING A BLINDED PEDERSEN COMMITMENT). For any algebraic adversary  $\mathcal{U}$  that on input of commitment parameters  $\vec{h}$  outputs blinded commitment parameters  $\vec{\Psi}'$ , a commitment  $\zeta_1$ , a partial vector  $\vec{L}'$  that has non- $\perp$  entries at index I a proof  $\pi = (\pi_{sdl}, \pi_{op})$  auxiliary information aux such that  $\pi = (\pi_{sdl}, \pi_{op})$ is the Fiat-Shamir transformation (see definition A.6) of the ANDproof (see lemma A.7) of the protocols described in definitions B.1 and B.6 along with group element representations for all group elements output by the adversary  $\mathcal{U}$ , there exists an extractor  $\mathcal{E}$  that on input of the adversary's output including all algebraic representations that were made during the output or to the Random Oracle, outputs a witness  $\gamma$  with probability  $1 - Q_h \frac{1+n}{p}$  (over the choice of the random oracle) and (if the former happens) additionally either a witness  $\vec{L}$ which is a partial opening to the commitment  $\zeta_1$  on the indices  $i \notin I$ , or a solution  $R, \vec{z}, \vec{z}'$  to the BINDING game of Pedersen commitments with probability  $1 - \frac{2}{p}$  over the choice of the random oracle.

PROOF. The extractor proceeds as follows. First, it calls the extractor from lemma B.4 on the blinded commitment parameters transcript  $\pi_{sdl}$  along with the commitment parameters  $\vec{h}$  and the blinded parameters  $\vec{\psi}$ . This extractor outputs a witness  $\gamma$  with probability  $1 - Q_h \cdot \frac{1+n}{p}$  as the adversary may query the RO  $Q_h$  many

times, so we union bound over the choice of the challenges for the  $\Sigma$ -protocol transcript.

The extractor then uses  $\gamma$  to transform the commitment  $\zeta_1$  into  $C = \zeta_1^{\frac{1}{\gamma}}$  and  $R_{op}$  from  $\pi_{op}$  into  $R'_{op} = R_{op}^{\frac{1}{\gamma}}$ . It transforms the algebraic representations by dividing by  $\gamma$ . It then runs the extractor from lemma B.10 to extract either a witness or a binding game solution. It outputs the outputs of the two extractors.

THEOREM 3.1 (EXTRACTOR FOR SHOWING PROCEDURE). For any algebraic adversary  $\mathcal{U}$  that, on input of commitment parameters  $pp_{GPC} = \vec{h}$ , it outputs an ACL public key pk along with a tuple  $(m, \sigma, \vec{L}', I_{op}, \bar{\pi})$  such that  $S\mathcal{H}_V(pk, m, \sigma, \vec{L}', I_{op}, \bar{\pi}) = 1$ , there exists an extractor  $\mathcal{E}$  that takes  $\mathcal{U}$ 's output  $(pk, (m, \sigma, \vec{L}', I_{op}, \bar{\pi}))$  along with all algebraic representations  $\mathcal{U}$  outputs as input, and it outputs a witness  $\gamma$  of the proof of same discrete logarithm with probability  $1 - Q_h \frac{1+n}{p}$  (over the choice of the random oracle) and (if the former happens) additionally either a witness  $\vec{L}$  which is a partial opening to the commitment  $\zeta_1$  on the indices  $i \notin I$ , or a solution  $R, \vec{z}, \vec{z}'$  to the **BINDING** game of Pedersen commitments with probability  $1 - \frac{2}{p}$  over the choice of the random oracle.

PROOF. This follows directly from theorem B.11. □

# C PROOFS OF OF LEMMAS 3.4 AND 3.7

Next, we propose a simplified variant of the restrictive blinding game from [1, 37] where we do not require that the adversary's output contains as many signatures as it requested signing sessions, but rather only a "mismatched" signature.

Definition 3.3 (Restrictive Blinding Game). We describe the restrictive blinding game for an adversary A:

- **Setup:** The challenger samples a key pair (pk, sk) according to  $\mathcal{G}(pp_{\mathbb{G}})$ . This uses the random oracle H<sub>1</sub> which allows the challenger to keep track of the discrete logarithms of its outputs. It outputs pk to the adversary.
- **Online Phase:** The adversary gets access to the following oracles
- **Oracle** H<sub>1</sub>: lazily samples  $w \leftarrow_{\$} \mathbb{Z}_p$  and returns  $g^w$ . Keeps a list of the hash inputs, w, and the output  $g^w$ .
- **Oracle**  $H_{pp}$ : lazily samples  $v \leftarrow_{\$} \mathbb{Z}_p$  and returns  $g^v$ . Keeps a list of the hash inputs v, and the output  $g^v$ .
- **Oracle** H<sub>3</sub>: lazily samples  $\varepsilon \leftarrow_{\$} \mathbb{Z}_p$  and returns  $\varepsilon$
- $\mathcal{R}_{\mathcal{S}}$ : Register new commitments and store them in list to ensure all commitments coming in  $\mathcal{S}_1$  queries were already registered.
- $S_1$ : runs the signing procedure  $S_1$  of ACL and returns its output along with a fresh session ID. Keeps a list of the commitment *C* and the random value rnd.
- $S_2$ : takes as input a session ID and a challenge *e*. If the session ID has never been used before in a  $S_2$  query, it runs the  $S_2$  procedure of ACL using the internal signer state corresponding to the session ID with *e* and returns its output.
- **Output Determination:**  $\mathcal{A}$  submits a list of message-sign ature pairs  $(m_i, \sigma_i)$  for  $i = 1..\ell$ . The game outputs 1 if there is a pair  $(m, \sigma = (\zeta, \zeta_1, ...))$  in the list such that the

signature is valid on the message *m* under the public key pk and it holds that there exists no closed signing session sid such that  $dlog_z \zeta = dlog_{z_{1,sid}} \zeta_1$ .

We define the advantage of the adversary in the game RB as

$$\operatorname{Adv}_{\operatorname{ACL}}^{\operatorname{RB}}(\mathcal{A}) \coloneqq \Pr[\operatorname{RB}_{\operatorname{ACL}}^{\mathcal{A}} = 1].$$

LEMMA 3.4 (RESTRICTIVE BLINDING). In the ROM and AGM, if DLP is  $(t, \varepsilon)$ -hard in  $\mathbb{G}$ , it holds that for any adversary running in time at most t, the advantage is at most

$$\operatorname{Adv}_{\operatorname{ACL}}^{\operatorname{RB}}(\mathcal{A}) \le 13 \cdot \varepsilon + \frac{4 \cdot Q_C + 31}{p} + \frac{1}{n}.$$

where  $Q_C$  is the number of registration queries made by the adversary.

PROOF. We use a similar proof technique to the one used in Theorem 4.1. in [35] (and full version [36]). We proceed in a series of games to gradually rule out bad event, reaching a game that we can reduce DLP if  $\mathcal A$  outputs a valid signature such that there exists no closed signing session sid satisfying  $dlog_z \zeta = dlog_{z_{1,sid}} \zeta_1$ .

We start by ruling out the first abort condition corresponding to the representation of the registered commitments C.

Game  $G_1$ . This is Game RB.

Let  $Q_C$  be the number of commitments  $\mathcal{A}$  registers. We define **BAD**<sub>1</sub> as the event in which  $\mathcal{A}$  registers a commitment  $C_i$ for  $j \in [Q_C]$  successfully using a proof  $\pi = (M, s_0, \dots, s_n)$  while  $T_{pp}[(C, M)] = \bot$ , where  $T_P$  is the table of the random oracle  $H_{reg}$ used in the Fiat-Shamir transformation of the protocol from definition 2.7. Intuitively, this corresponds to  $\mathcal{A}$  registering a commitment  $C_i$  successfully without having made a hash query beforehand.

Game  $G_2$ . This game is identical to Game  $G_1$ , except that it aborts if Event BAD1 occurs.

Claim C.1.

$$\Pr[BAD_1] \leq \frac{Q_C}{p}.$$

**PROOF.** Outputting a valid proof  $\pi = (M, s_0, \ldots, s_n)$  for a commitment *C* implies that the equation  $C^c \cdot \prod_{i \in [\llbracket n \rrbracket} h_i^{s_i} = M$ , where  $c = H_{reg}(C, M)$ . Thus, outputting a valid proof without querying  $H_{reg}$  beforehand to acquire *c* is equivalent to guessing *c*. However, sine  $H_{reg}$  is modeled as a random oracle, the value *c* is chosen uniformly at random, and the probability of  ${\mathcal A}$  guessing this value correctly is at most  $\frac{1}{p}$ . Since  $\mathcal{A}$  registers up to  $Q_C$  values, the probability that it successfully registers some commitment without having made a corresponding hash query is at most  $\frac{Q_C}{p}$ . 

By this claim, it holds  $\operatorname{Adv}_{G_2}^{\mathcal{A}} \geq \operatorname{Adv}_{G_1}^{\mathcal{A}} - \frac{Q_C}{p}$ . As  $\operatorname{BAD}_1$  does not occur, we assume in the following that for each valid commitment-proof tuple  $(C, \pi = (M, s_0, \dots, s_n))$  the adversary outputs, it holds that  $T_P \neq \bot$ .

Let  $BAD_2$  be the event in which  $\mathcal{A}$  successfully registers a commitment  $C_j$  for  $j \in [Q_C]$  satisfying the condition

$$\bigvee_{\in [Q_C]} y_{[C_j]} + \sum_{\text{sid} \in Q_S} c_{\text{sid}} \cdot a_{\text{sid}[C_j]} \neq 0,$$

where  $Q_S$  is the number of open signing sessions at the time  $\mathcal{A}$ registers  $C_i$ , and  $c_{sid}$  is the random chosen by the signer to compute  $a_{sid}$ . Intuitively, this condition means that  $\mathcal{A}$  uses y or some  $a_{sid}$ (or both) in the representation of some commitment  $C_i$  such that they do not cancel each other out.

Game  $G_3$ . This game is identical to  $G_2$ , except that it aborts if event BAD<sub>2</sub> occurs.

Claim C.2.

$$\Pr[\mathbf{BAD}_2] \le \mathbf{Adv}_{\mathbf{DL}}^{\mathcal{R}1} + \frac{1}{p}.$$

PROOF. It follows directly from Claim E.1 in [35].

By the claim, we have  $\operatorname{Adv}_{G_3}^{\mathcal{A}} \ge \operatorname{Adv}_{G_2}^{\mathcal{A}} - \operatorname{Adv}_{DL}^{\mathcal{R}_1} - \frac{1}{p}$ . By this claim, we assume that  $\mathcal{A}$  wins  $G_3$  and  $\operatorname{BAD}_2$  does not

occur. However,  $\mathcal{A}$  may still use  $a_{sid}$  or y components in the representation of C without triggering  $BAD_2$  in a way that prevents  $\mathcal R$  from functioning properly. Specifically, if  $\mathcal A$  uses the group element  $a_{sid}$  or  $b_{2,sid}$  of the signing session sid in the representation of some commitment C, but it never closes sid, this may lead to an issue for  $\mathcal{R}$ , because it won't be able to compute  $dlog_a C$  without knowing the opening of  $a_{sid}$  or  $b_{sid}$ , i.e.  $(r_{sid}, c_{sid})$  and  $(s_{2,sid}, d_{sid})$ , respectively. Note that this does not cause an issue for the current game hop, because it is straight forward to known the opening of all group elements of any signing session sid since the signer oracles are simulated by the reduction. However, this is problematic later in the reduction from Game OMMIM to Game OMUF, because  ${\cal R}$ uses the challenger's prover oracles to simulate the signer oracles for  $\mathcal{A}$ . Consequently, the reduction will have access to the opening of a group element  $a_{\mathsf{sid}}$  or  $b_{\mathsf{sid}}$  2 iff the adversary makes  $\mathcal{S}$  2 query for sessions sid. Therefore, it is crucial to ensure that no  $a_{sid}$  or  $b_{2,sid}$  is used in the representation of any commitment C if sid will not be closed. To capture this, we define BAD<sub>3</sub> as the event that occurs if the equation

$$\bigvee_{i \in [Q_C]} (y_{[C_j]} + \sum_{\mathsf{sid} \in \vec{I}_{op}} c_{\mathsf{sid}} \cdot a_{\mathsf{sid}[C_j]} = -\sum_{\mathsf{sid} \in \vec{I}_{cl}} c_{\mathsf{sid}} \cdot a_{\mathsf{sid}[C_j]}).$$

Game  $G_4$ . This game is identical to  $G_3$ , except that it aborts if event BAD3 occurs.

Claim C.3.

j

$$\Pr[\mathbf{BAD}_3] \le \frac{Q_C}{p}.$$

PROOF. Identical to Claim E.2 in [35].

Consequently, we have  $\operatorname{Adv}_{G_4}^{\mathcal{A}} \ge \operatorname{Adv}_{G_3}^{\mathcal{A}} - \frac{Q_C}{p}$ . Next, we rule out the usage of any h or  $b_{2,sid}$  component in the

representation of  $C_j$  for sid  $\in Q_S$  and  $j \in Q_C$ . We define Event BAD<sub>4</sub> as the event that occurs if

$$\bigvee_{j \in [Q_C]} (h_{[C_j]} + \sum_{\mathsf{sid} \in Q_S} s_{2,\mathsf{sid}} \cdot b_{2,\mathsf{sid}}_{[C_j]}) \neq 0$$

holds, where  $C_i$  are the commitments registered by  $\mathcal{A}$ . This condition means that  $\mathcal{A}$  uses h or some  $b_{2,sid}$  (or both) in the representation of some commitment  $C_i$  such that they do not cancel each other out.

*Game*  $G_5$ . This game is identical to  $G_4$ , except that it aborts if event BAD<sub>4</sub> occurs.

Claim C.4.

$$\Pr[\mathbf{BAD}_4] \le \mathbf{Adv}_{\mathbf{DL}}^{\mathcal{R}2} + \frac{1}{p}.$$

PROOF. Identical to Claim E.3 in [35].

Per this claim, we have  $\operatorname{Adv}_{G_5}^{\mathcal{A}} \ge \operatorname{Adv}_{G_4}^{\mathcal{A}} - \operatorname{Adv}_{\operatorname{DL}}^{\mathcal{R}2} - \frac{1}{p}$ . Next, we prevent  $\mathcal{A}$  from using  $b_{2,\operatorname{sid}}$  from an open signing

Next, we prevent  $\mathcal{A}$  from using  $b_{2,sid}$  from an open signing session sid in the representation of some  $C_j$  for  $j \in [Q_C]$  without triggering **BAD**<sub>4</sub>.

Define BAD<sub>5</sub> as the event that occurs if

$$\bigvee_{j \in [Q_C]} (h_{[C_j]} + \sum_{\mathsf{sid} \in \vec{I}_{cl}} s_{2,\mathsf{sid}} \cdot b_{2,\mathsf{sid}[C_j]} = -\sum_{\mathsf{sid} \in \vec{I}_{op}} s_{2,\mathsf{sid}} \cdot b_{2,\mathsf{sid}[C_j]}).$$

This condition means that  $\mathcal{A}$  uses *h* or some  $b_{2,\text{sid}}$  (or both) in the representation of some commitment  $C_j$  such that they do not cancel each other out.

*Game*  $G_6$ . This game is identical to  $G_5$ , except that it aborts if event BAD<sub>5</sub> occurs.

Claim C.5.

$$\Pr[\mathbf{BAD}_5] \le \frac{Q_C}{p}.$$

PROOF. The same as Claim E.4 in [35].

By the claim, it holds  $\operatorname{Adv}_{G_6}^{\mathcal{A}} \ge \operatorname{Adv}_{G_5}^{\mathcal{A}} - \frac{Q_C}{p}$ . Let Event BAD<sub>6</sub> be the event that occurs if

$$\bigvee_{i \in [Q_C]} (z_{[C_j]} + \sum_{\mathsf{sid} \in Q_S} d_{\mathsf{sid}} \cdot b_{2,\mathsf{sid}[C_j]} \neq 0)$$

holds. This condition is equivalent to  $\mathcal{A}$  using *z* or some  $b_{2,sid}$  (or both) in the representation of some commitment  $C_j$  without canceling each other out.

Game  $G_7$ . This game is identical to  $G_6$ , except that it aborts if event BAD<sub>6</sub> occurs.

Claim C.6.

$$\Pr[\mathbf{BAD}_6] \le \mathbf{Adv}_{\mathbf{DL}}^{\mathcal{R}3} + \frac{1}{p}.$$

PROOF. The same as Claim E.5 in [35].

It follows that  $\operatorname{Adv}_{\mathbf{G}_7}^{\mathcal{A}} \geq \operatorname{Adv}_{\mathbf{G}_6}^{\mathcal{A}} - \operatorname{Adv}_{\mathrm{DL}}^{\mathcal{R}_3} - \frac{1}{p}$ .

Similar to what we did in Games  $G_4$  and  $G_6$ , we rule out using  $b_{2,sid}$  from a signing session sid that is never closed in the representation of some commitment  $C_j$  without triggering BAD<sub>6</sub>.

Let Event BAD<sub>7</sub> be the event that occurs if

$$\bigvee_{j \in [Q_C]} z_{[C_j]} + \sum_{\mathsf{sid} \in \vec{I}_{op}} d_{\mathsf{sid}} \cdot b_{2,\mathsf{sid}[C_j]} = -\sum_{\mathsf{sid} \in \vec{I}_{cl}} d_{\mathsf{sid}} \cdot b_{2,\mathsf{sid}[C_j]}$$

holds.

*Game*  $G_8$ . This game is identical to  $G_7$ , except that it aborts if event BAD<sub>7</sub> occurs.

Claim C.7.

$$\Pr[\mathbf{BAD}_7] \le \frac{Q_C}{p}$$

п

PROOF. Identical to Claim E.6 in [35].

It follows that  $\operatorname{Adv}_{G_8}^{\mathcal{A}} \geq \operatorname{Adv}_{G_7}^{\mathcal{A}} - \frac{Q_C}{p}$ . Let open<sub>j</sub> be a vector of all signing sessions identifiers that are

Let open<sub>j</sub> be a vector of all signing sessions identifiers that are still open at the time  $\mathcal{A}$  registers a commitment  $C_j$  for  $j \in [Q_C]$ , ordered by the time the sessions were opened. Let Event **BAD**<sub>8</sub> be the event that occurs if

$$\bigvee_{i \in [Q_C]} \sum_{\text{sid} \in \text{open}_j} u_{3,\text{sid}} \cdot b_{1,\text{sid}[C_j]} \neq 0,$$

 $\Pr[BAD_8] \le Adv_{DL}^{\mathcal{R}4} + \frac{3}{n} + \frac{1}{n}.$ 

holds, where  $u_{3,\text{sid}} \coloneqq \text{dlog}_q b_{1,\text{sid}}$ .

Claim C.8.

By this claim, it holds that  $Adv_{G_8}^{\mathcal{A}} \geq Adv_{G_7}^{\mathcal{A}} - Adv_{DL}^{\mathcal{R}4} - \frac{3}{p} - \frac{1}{n}$ .

COROLLARY C.9. Given a representation of a successfully registered commitment C,  $\mathcal{R}$  can efficiently compute  $\operatorname{dlog}_a C$ .

PROOF. If  $\mathcal{A}$  wins  $G_8$ , non of the abort conditions of the previous games (i.e.,  $BAD_1 - BAD_8$ ) was triggered, except with negligible probability. In this case, the representations of all registered commitments C do not contain any group elements with unknown discrete logarithms to  $\mathcal{R}$ , such as the group elements of the challenger's public key or the group elements  $\mathcal{A}$  receiver via  $S_1$  queries. Thus,  $\mathcal{R}$  can efficiently compute  $dlog_q C$ .

Next, we address the second abort condition, which relates to the representation of the group element  $\zeta$ .

Let BAD<sub>9</sub> be the event in which  $\mathcal{A}$  outputs a valid signature  $(\zeta, \zeta_1, \rho, \omega, \sigma_1, \sigma_2, \mu, \delta)$  for a message *m*, while  $\mathsf{T}_3[(\zeta, \zeta_1, g^\rho \cdot y^\omega, g^{\sigma_1} \cdot \zeta_1^\delta, h^{\sigma_2} \cdot \zeta_2^\delta, z^\mu \cdot \zeta^\delta, m)] = \bot$ . This corresponds to  $\mathcal{A}$  outputting a valid signature without making a corresponding hash query to  $\mathsf{H}_3$  beforehand.

Game  $G_9$ . This game is the same as  $G_8$  except that it aborts if Event BAD<sub>9</sub> occurs.

Claim C.10.

$$\Pr[\mathsf{BAD}_9] \le \mathrm{Adv}_{\mathsf{G}_8}^{\mathcal{A}} + \frac{1}{p}.$$

PROOF. A valid signature must satisfy the verification equation  $\omega + \delta = H_3(\zeta, \zeta_1, g^{\rho} \cdot y^{\omega}, g^{\sigma_1} \cdot \zeta_1^{\delta}, h^{\sigma_2} \cdot \zeta_2^{\delta}, z^{\mu} \cdot \zeta^{\delta}, m)$ . However, outputting a valid signature without querying H<sub>3</sub> beforehand is equivalent to guessing the output of H<sub>3</sub>, i.e.  $\omega + \delta$ , on input  $(\zeta, \zeta_1, g^{\rho}, y^{\omega}, g^{\sigma_1} \cdot \zeta_1^{\delta}, h^{\sigma_2} \cdot \zeta_2^{\delta}, z^{\mu} \cdot \zeta^{\delta}, m)$ . As H<sub>3</sub> is modeled as a random oracle, its output is chosen uniformly at random from  $\mathbb{Z}_p$  and the probability that  $\mathcal{A}$  succeeds in guessing its output is at most  $\frac{1}{p}$ .  $\Box$ 

:c

By the claim, it holds  $\operatorname{Adv}_{G_9}^{\mathcal{A}} \ge \operatorname{Adv}_{G_8}^{\mathcal{A}} - \frac{1}{p}$ .

PROOF. Identical to Claim E.11 in [35].

Define Event **BAD**<sub>10</sub> as the event that occurs if  $\mathcal{A}$  outputs a valid signature  $(\zeta, \zeta_1, \rho, \omega, \sigma_1, \sigma_2, \mu, \delta)$  for a message *m* with a corresponding H<sub>3</sub> hash query  $(\zeta, \zeta_1, \alpha, \beta_1, \beta_2, \eta, m)$ , and it holds  $z^{z_{\lfloor \zeta \rfloor}} \neq \zeta$ . That is, the event occurs if the representation of  $\zeta$  contains any group element other than *z*.

*Game*  $G_{10}$ . This game is identical to  $G_9$ , except that it aborts if  $BAD_{10}$  occurs.

Claim C.11.

$$\Pr[\mathbf{BAD}_{10}] \le \mathbf{Adv}_{\mathbf{DL}}^{\mathcal{R}5} + 2 \cdot \mathbf{Adv}_{\mathbf{DL}}^{\mathcal{R}6} + \frac{6}{p}.$$

By this claim, we have  $\operatorname{Adv}_{G_{10}}^{\mathcal{A}} \geq \operatorname{Adv}_{G_9}^{\mathcal{A}} - \operatorname{Adv}_{DL}^{\mathcal{R}5} - 2 \cdot \operatorname{Adv}_{DL}^{\mathcal{R}6} - \frac{6}{p}$ , and therefore, we assume  $\mathcal{A}$  wins  $G_{10}$  and  $\zeta$  is computed honestly, i.e. its representation contains only a *z* component.

COROLLARY C.12. For any valid signature  $(\zeta, \zeta_1, \rho, \omega, \sigma_1, \sigma_2, \mu, \delta)$ with a corresponding H<sub>3</sub> hash query  $(\zeta, \zeta_1, \alpha, \beta_1, \beta_2, \eta, m)$ ,  $\mathcal{R}$  can use the representation of  $\zeta$  submitted along the hash query to efficiently compute dlog<sub>z</sub> $\zeta$ .

**PROOF.** If  $\mathcal{A}$  wins  $\mathbf{G}_{10}$ , the abort conditions  $\mathbf{BAD}_9$  and  $\mathbf{BAD}_{10}$  cannot be triggered, except with probability  $\frac{7}{p} + \mathbf{Adv}_{\mathbf{DL}}^{\mathcal{R}5} + 2 \cdot \mathbf{Adv}_{\mathbf{DL}}^{\mathcal{R}6}$ In this case, the representation of the group element  $\zeta$  of every hash query corresponding to some valid signature only consists solely of a *z* component, i.e.  $\zeta = g^{z[\zeta]}$ . Consequently,  $\mathcal{R}$  can efficiently compute  $\mathrm{dlog}_z\zeta$ , except with negligible probability.

Let Event **BAD**<sub>11</sub> be the event that occurs if  $\mathcal{A}$  outputs a valid signature  $(\zeta, \zeta_1, \rho, \omega, \sigma_1, \sigma_2, \mu, \delta)$  for a message *m* with a corresponding H<sub>3</sub> hash query  $(\zeta, \zeta_1, \alpha, \beta_1, \beta_2, \eta, m)$  satisfying  $dlog_z z_{1,sid} \neq dlog_\zeta \zeta_1$  for all signing session sid.

Game  $G_{11}$ . This game is identical to  $G_{10}$ , except that it aborts if  $BAD_{11}$  occurs.

Claim C.13.

$$\Pr[\mathsf{BAD}_{11}] \le \mathsf{Adv}_{\mathsf{DL}}^{\mathcal{R}7} + \mathsf{Adv}_{\mathsf{DL}}^{\mathcal{R}8} + 4 \cdot \mathsf{Adv}_{\mathsf{DL}}^{\mathcal{R}9} + \frac{18}{p}$$

PROOF. The same as Claim E.11 in [35].

By this claim, we have  $\operatorname{Adv}_{G_{11}}^{\mathcal{A}} \geq \operatorname{Adv}_{G_{10}}^{\mathcal{A}} - \operatorname{Adv}_{DL}^{\mathcal{R}_7} - \operatorname{Adv}_{DL}^{\mathcal{R}_8} - 4 \cdot \operatorname{Adv}_{DL}^{\mathcal{R}_9} - \frac{18}{p}$ . Therefore, It must hold that each valid signature output by  $\mathcal{A}$  is linked to some signing session sid by satisfying  $\operatorname{dlog}_z z_{1,\text{sid}} = \operatorname{dlog}_{\zeta} \zeta_1$ .

COROLLARY C.14. For any valid signature  $(\zeta, \zeta_1, \rho, \omega, \sigma_1, \sigma_2, \mu, \delta)$ with a corresponding H<sub>3</sub> hash query  $(\zeta, \zeta_1, \alpha, \beta_1, \beta_2, \eta, m)$ , let  $\gamma := d\log_z \zeta$ . Then there exists a signing session sid such that  $\zeta_1^{1/\gamma} = z_{1,sid}$ , hence  $d\log_\zeta \zeta_1^{1/\gamma} = rnd_{sid}$ .

PROOF. If  $\mathcal{A}$  wins G<sub>11</sub>, the abort condition BAD<sub>11</sub> was not triggered, except with negligible probability. It follows that any valid signature satisfies  $dlog_z z_{1,sid} = dlog_\zeta \zeta_1$  for some signing session sid. Define Event **BAD**<sub>11</sub>' as the event that occurs if  $\mathcal{A}$  outputs a valid signature  $(\zeta, \zeta_1, \rho, \omega, \sigma_1, \sigma_2, \mu, \delta)$  with a corresponding H<sub>3</sub> hash query  $(\zeta, \zeta_1, \alpha, \beta_1, \beta_2, \eta, m)$  such that for all signing sessions sid it holds  $\operatorname{dlog}_{z\zeta} = \operatorname{dlog}_{z_{sid}1}\zeta_1$ .

Corollary C.15.

$$\Pr[\mathsf{BAD}_{11}'] \le \mathsf{Adv}_{\mathsf{DL}}^{\mathcal{R}7} + \mathsf{Adv}_{\mathsf{DL}}^{\mathcal{R}8} + 4 \cdot \mathsf{Adv}_{\mathsf{DL}}^{\mathcal{R}9} + \frac{18}{p}$$

PROOF. We know that  $\Pr[BAD_{11}] \leq Adv_{DL}^{\mathcal{R}7} + Adv_{DL}^{\mathcal{R}8} + 4 \cdot Adv_{DL}^{\mathcal{R}9} + \frac{18}{p}$ , thus, it suffices to show that  $BAD_{11} \Leftrightarrow BAD_{11}'$ . Let  $\gamma \coloneqq d\log_2 \zeta$ . It follows that  $d\log_z z_{1,sid} = d\log_z \zeta \zeta_1 \rightleftharpoons v_0$ , hence  $\zeta_1 = z^{v_0 \cdot \gamma} = z_{1,sid}^{\gamma}$ . It follows that  $d\log_z z_{1,sid} = d\log_\zeta \zeta_1 \Leftrightarrow d\log_z \zeta = d\log_{\zeta_{1,sid}} \zeta_1$ .

Due to Corollary C.15, we have

$$\begin{aligned} \operatorname{Adv}_{\operatorname{ACL}}^{\operatorname{RB}}(\mathcal{A}) &\leq \frac{4 \cdot Q_C + 31}{p} + \frac{1}{n} + \operatorname{Adv}_{\operatorname{DL}}^{\mathcal{R}1} + \operatorname{Adv}_{\operatorname{DL}}^{\mathcal{R}2} + \operatorname{Adv}_{\operatorname{DL}}^{\mathcal{R}3} \\ + \operatorname{Adv}_{\operatorname{DL}}^{\mathcal{R}4} + \operatorname{Adv}_{\operatorname{DL}}^{\mathcal{R}5} + 2 \cdot \operatorname{Adv}_{\operatorname{DL}}^{\mathcal{R}6} + \operatorname{Adv}_{\operatorname{DL}}^{\mathcal{R}7} + \operatorname{Adv}_{\operatorname{DL}}^{\mathcal{R}8} + 4 \cdot \operatorname{Adv}_{\operatorname{DL}}^{\mathcal{R}9}, \\ & \text{hence} \end{aligned}$$

$$\operatorname{Adv}_{\operatorname{ACL}}^{\operatorname{RB}}(\mathcal{A}) \le 13 \cdot \varepsilon + \frac{4 \cdot Q_C + 31}{p} + \frac{1}{n}.$$

Definition C.16 (2-to-1 tagging game). [Game  $\ell$ -2-to-1-Tagging ( $\ell$ -TTOT)] For a positive integer  $\ell \in \mathbb{Z}^+$ , we define the game  $\ell$ -TTOT for the identification scheme  $ID_{ACL} := (\mathcal{G}, \mathcal{P}, \mathcal{V})$  and an adversary  $\mathcal{A}$  as follows.

- **Initialization.** The challenger samples a key pair (pk, sk)  $\leftarrow_{\$} \mathcal{G}(pp_{\mathbb{G}})$ . It outputs pk to the adversary  $\mathcal{A}$ .
- **Online Phase.** The adversary gets to interact with prover oracles  $\mathcal{P}_1$  and  $\mathcal{P}_2$  and verifier oracles  $\mathcal{V}_1$ ,  $\mathcal{V}_2$ .
- **Output Determination.** The game outputs 1 there are two accepting sessions  $i_1, i_2$  with the verifier, such that there exists a  $\mathcal{P}_2$  invocation j, it holds that  $\zeta_{i_1,i} = z_{1,j} = \zeta_{i_2,i}$ . We call these two verifier sessions the *special sessions*.

We define the advantage of an adversary  $\mathcal{A}$  in winning the game for the ACL identification scheme ID<sub>ACL</sub> as

$$\operatorname{Adv}_{\mathsf{ID}_{\mathsf{ACL}}}^{\ell\operatorname{-TTO}}(\mathcal{A}) \coloneqq \Pr[\ell\operatorname{-TTO}^{\mathcal{A}}_{\mathsf{ID}_{\mathsf{ACL}}} = 1].$$

LEMMA C.17 (2-TO-1 TAGGING (ADAPTED FROM LEMMA 3.5 IN [35])). There exists an adversary  $\mathcal{R}$  against the discrete logarithm problem such that for any algebraic adversary  $\mathcal{A}$  it holds that

$$\operatorname{Adv}^{\operatorname{DLP}}(\mathcal{R}) \geq \frac{1}{2}\operatorname{Adv}^{\ell\operatorname{-TTOT}}_{\operatorname{ID}_{\operatorname{ACL}}}(\mathcal{A}) - \frac{1}{p}.$$

PROOF. The proof is identical to the proof of Lemma 3.5 in [35] as their proof only uses the fact that there are two special sessions with the same tag.

*Definition 3.6 (2-to-1 Game).* We describe the *2-to-1* game  $\ell$ -TTO for an adversary  $\mathcal{A}$ :

**Setup:** The challenger samples a key pair (pk, sk) according to  $\mathcal{G}(pp_{\mathbb{G}})$ . This uses the random oracle H<sub>1</sub> which allows the challenger to keep track of the discrete logarithms of its outputs. It outputs pk to the adversary.

- **Online Phase:** The adversary gets access to the following oracles
- **Oracle** H<sub>1</sub>: lazily samples  $w \leftarrow_{\$} \mathbb{Z}_p$  and returns  $g^w$ . Keeps a list of the hash inputs, w, and the output  $g^w$ .
- **Oracle** H<sub>pp</sub>: lazily samples  $v \leftarrow_{\$} \mathbb{Z}_p$  and returns  $g^v$ . Keeps a list of the hash inputs v, and the output  $g^v$ .
- **Oracle** H<sub>3</sub> lazily samples  $\varepsilon \leftarrow_{\$} \mathbb{Z}_p$  and returns  $\varepsilon$
- $S_1$ : runs the signing procedure  $S_1$  of ACL and returns its output along with a fresh session ID. Keeps a list of the commitment *C* and the
- $S_2$ : takes as input a session ID and a challenge *e*. If the session ID has never been used before in a  $S_2$  query, it runs the  $S_2$  procedure of ACL using the internal signer state corresponding to the session ID with *e* and returns its output.
- **Output Determination:** The adversary submits a list of message-signature pairs  $(m_i, \sigma_i)$  for  $i = 1..\ell$ . The game outputs 1 if there are two messages  $m_1, m_2$  and two signatures  $(\zeta^{(1)}, \zeta_1^{(1)}, \rho^{(1)}, \omega^{(1)}, \sigma_1^{(1)}, \sigma_2^{(1)}, \mu^{(1)}, \delta^{(1)})$  and  $(\zeta^{(2)}, \zeta_1^{(2)}, \rho^{(2)}, \omega^{(2)}, \sigma_2^{(2)}, \mu^{(2)}, \delta^{(2)})$  im the list such that the signatures are valid on the respective messages under the public key pk and it holds that there exists a single closed signing session such that  $dlog_z \zeta^{(1)} = dlog_{C\cdot g^{rnd}} \zeta_1^{(1)}$  and  $dlog_z \zeta^{(2)} = dlog_{C\cdot g^{rnd}} \zeta_1^{(2)}$ .

We define the advantage of the adversary in the game  $\ell$ -TTO as

$$\operatorname{Adv}_{\operatorname{ACl}}^{\ell-\operatorname{TTO}}(\mathcal{A}) \coloneqq \Pr[\ell-\operatorname{TTO}_{\operatorname{ACl}}^{\mathcal{A}} = 1].$$

LEMMA 3.7 (2-TO-1 BLINDING). In the ROM and AGM, if DLP is  $(t, \varepsilon)$ -hard in  $\mathbb{G}$ , it holds that for any adversary running in time at most t, the advantage is at most

$$\operatorname{Adv}_{\operatorname{ACL}}^{\ell\operatorname{-TTO}}(\mathcal{A}) \leq 7 \cdot \varepsilon + \frac{20}{p}$$

PROOF. Assume there is an algebraic adversary  $\mathcal{A}$  that runs and wins experiment  $\ell$ -TTO. It follows that  $\mathcal{A}$  provides two valid signatures  $(\zeta^{(1)}, \zeta_1^{(1)}, \rho^{(1)}, \omega^{(1)}, \sigma_1^{(1)}, \sigma_2^{(1)}, \mu^{(1)}, \delta^{(1)})$  and  $(\zeta^{(2)}, \zeta_1^{(2)}, \rho^{(2)}, \omega^{(2)}, \sigma_2^{(2)}, \mu^{(2)}, \delta^{(2)})$  satisfying the equalities  $\operatorname{dlog}_{z}\zeta^{(1)} = \operatorname{dlog}_{C \cdot g^{\mathrm{rnd}}}\zeta_1^{(1)}$  and  $\operatorname{dlog}_{z}\zeta^{(2)} = \operatorname{dlog}_{C \cdot g^{\mathrm{rnd}}}\zeta_1^{(2)}$ . Using the same game hop from the proof of Lemma 3.4 (G<sub>2</sub>-G<sub>11</sub>), we rule out the abort events of these Games. Consequently, Corollaries C.9, C.12, and C.14 hold, and  $\mathcal{R}$ 9 can efficiently compute  $\operatorname{dlog}_{g}C$  for each registered commitment C,  $\operatorname{dlog}_{z}\zeta$ , and  $\operatorname{dlog}_{z_{1},\mathrm{sid}}\zeta_1$  of both hash queries linked to the signatures  $\mathcal{A}$  outputs, for some signing sessions sid.

When  $\mathcal{R}$  is executed with game  $\ell$ -TTOT, it takes pk = (h, y, z) as input and gets access to the oracles  $\mathcal{P}_1, \mathcal{P}_2, \mathcal{V}_1$ , and  $\mathcal{V}_2$ . Then, it invokes  $\mathcal{A}$  with input pk and simulates Game  $\ell$ -TTO for  $\mathcal{A}$  by providing access to the following oracles:

- *R<sub>S</sub>*(*C*, *π*). If *π* is a valid proof of knowledge of an opening for *C* and *C* ∉ registered, store *C* in registered and output 1, 0 otherwise.
- S1(C). Output ⊥ if C ∉ registered. Query the challenger's oracle (sid, a<sub>sid</sub>, b<sub>1,sid</sub>, b<sub>2,sid</sub>, rnd<sub>sid</sub>) ←<sub>\$</sub> 𝒫<sub>1</sub>(⊥), store sid in open, compute rnd'<sub>sid</sub> := rnd<sub>sid</sub> dlog<sub>g</sub>C (this is possible due to Corollary C.9). Output (sid, a<sub>sid</sub>, b<sub>1,sid</sub>, b<sub>2,sid</sub>, rnd'<sub>sid</sub>).

- S2(sid, e<sub>sid</sub>). Output ⊥ if sid ∉ open. Store sid in closed, query the challenger's oracle (c<sub>sid</sub>, d<sub>sid</sub>, r<sub>sid</sub>, s<sub>1,sid</sub>, s<sub>2,sid</sub>) ← P<sub>2</sub>(sid, e<sub>sid</sub>), and output (c<sub>sid</sub>, d<sub>sid</sub>, r<sub>sid</sub>, s<sub>1,sid</sub>, s<sub>2,sid</sub>).
- $H_1(Q)$ . Via lazy sampling.
- H<sub>3</sub>(Q). If T<sub>3</sub>[Q] ≠ ⊥, output T<sub>3</sub>[Q]. Otherwise, parse (ζ, ζ<sub>1</sub>, α, β<sub>1</sub>, β<sub>2</sub>, η, m), compute γ := dlog<sub>z</sub>ζ using the representation of ζ if possible (if this query is used later to generate a signature, this is possible due to Corollary C.12), otherwise respond to the query with a random value. Compute z'<sub>1</sub> := ζ<sub>1</sub><sup>1/γ</sup>, and find a signing session sid such that z<sub>1,sid</sub> = z'<sub>1</sub> and set rnd\* = rnd<sub>sid</sub> (if this query is used later to generate a signature, this is possible due to Corollary C.14). If no such signing session exists, respond with a uniformly random value. Next, set A := α, B<sub>1</sub> := β<sub>1</sub><sup>1/γ</sup>, B<sub>2</sub> := β<sub>2</sub><sup>1/γ</sup>, make a query (vid, ε<sub>vid</sub>) ←<sub>\$</sub> V<sub>1</sub>(A, B<sub>1</sub>, B<sub>2</sub>, rnd\*). Finally, set T<sub>V</sub>[Q] := vid, T<sub>3</sub>[Q] := ε, and output ε.
- H<sub>pp</sub>(Q). Via lazy sampling.

When  $\mathcal{A}$  outputs a signature  $(\zeta, \zeta_1, \omega, \rho, \sigma_1, \sigma_2, \delta, \mu)$  satisfying the condition  $\operatorname{dlog}_z \zeta \neq \operatorname{dlog}_{z_{\operatorname{sid}1}}\zeta_1$  for all signing sessions sid,  $\mathcal{R}$ retrieves vid  $\leftarrow \mathsf{T}_3[(\zeta, \zeta_1, g^{\rho} \cdot y^{\omega}, g^{\sigma_1} \cdot \zeta_1^{\delta}, h^{\sigma_2} \cdot \zeta_2^{\delta}, z^{\mu} \cdot \zeta^{\delta})]$  for  $\zeta_2 \coloneqq \zeta/\zeta_1$ . If vid  $= \bot$ ,  $\mathcal{R}$  aborts and outputs  $\bot$ . This abort event does not occur because we know from game G<sub>9</sub> that each signature has a corresponding hash query, and the responses of such queries were generated via  $\mathcal{V}$  queries., hence each query is linked to a valid verifier session vid. Otherwise, it closes the session vid by sending  $(\omega, \delta, \rho, \sigma_1/\gamma, \sigma_2/\gamma)$ .

Analysis. First, we note that  $\mathcal{R}$ 's simulation for the signer's oracles is perfectly indistinguishable from an honest run of the protocol. To see this, note that the only difference between the prover's oracles of the challenger and the honest signer in ACL is the way  $z_1$  is computed. More specifically,  $z_1 = g^{\text{rnd}}$  in  $\text{ID}_{\text{ACL}}$  while  $z_1 = g^{\text{rnd}} \cdot C$  in ACL. To avoid this inconsistency,  $\mathcal{R}$  forwards an updated  $\text{rnd}' = \text{rnd} - \text{dlog}_g C$  to  $\mathcal{A}$  ensuring that  $g^{\text{rnd}'} \cdot C = g^{\text{rnd}} \cdot C^{-1} \cdot C = g^{\text{rnd}}$ . The resulting  $z_1$  (if computed honestly) at the adversary side would be identical to  $z_1$  at the prover side, hence forwarding the rest of the values from the prover to  $\mathcal{A}$  would preserve the correctness and ensure a perfect simulation.

Second, we note that, if the reduction  $\mathcal{R}$  does not abort and  $\mathcal{A}$  outputs a valid signature allowing it to win the RB lemma,  $\mathcal{R}$  would use this signature to close a verifier session successfully and satisfying the condition required to win the  $\ell$ -TTOT game. Once  $\mathcal{A}$  outputs a valid signature  $(\zeta, \zeta_1, \omega, \rho, \sigma_1, \sigma_2, \delta, \mu)$  satisfying the condition  $\operatorname{dlog}_z \zeta \neq \operatorname{dlog}_{z_{\operatorname{sid}}} \zeta_1$  for all signing sessions sid,  $\mathcal{R}$  searches for the verifier session vid that was opened to generate the challenge  $\varepsilon$  for this signature by searching  $\operatorname{Tv}[(\zeta, \zeta_1, g^{\rho} \cdot y^{\omega}, g^{\sigma_1} \cdot \zeta_1^{\delta}, h^{\sigma_2} \cdot \zeta_2^{\delta}, z^{\mu} \cdot \zeta^{\delta}, m)]$ . Then, it sets  $\omega' \coloneqq \omega, \delta' \coloneqq \delta, \rho' \coloneqq \rho, \sigma'_1 \coloneqq \sigma/\gamma, \sigma'_2 \coloneqq \sigma 2/\gamma)$  and closes the session vid by sending (vid,  $\omega', \rho', \delta', \sigma'_1, \sigma'_2$ ) in a  $\mathcal{V}_2$  query. The verifier outputs 1 iff the values  $(A, B_1, B_2, \operatorname{rnd}^*)$  and  $(\omega', \rho', \delta', \sigma'_1, \sigma'_2)$  satisfy the equalities  $\varepsilon = \omega + \delta, \alpha' = g^{\rho'} \cdot y^{\omega'}, B_1 = g^{\sigma_1} \cdot z_1^{\delta'}, B_2 = h^{\sigma'_2} \cdot z_2^{\delta'}$ , where  $z_1 = g^{\operatorname{rnd}^*}$  and  $z_2 \coloneqq z_1/z$ . Indeed, these equalities hold because

ω' +δ' = ε follows directly from the validity of the signature since ω = ω', δ = δ', and ε = H<sub>3</sub>(ζ, ζ<sub>1</sub>, α, β<sub>1</sub>, β<sub>2</sub>, η, m),

Foteini Baldimtsi, Julia Kastner, Julian Loss, and Omar Renawi

- α' = g<sup>ρ'</sup> · y<sup>ω'</sup> follows directly from the validity of the signature since α = α', ρ = ρ', and ω = ω'.
  Since the signature is valid, it holds that β<sub>1</sub> = g<sup>σ<sub>1</sub></sup> · ζ<sub>1</sub><sup>δ</sup> ⇒ β<sub>1</sub><sup>1/γ</sup> = (g<sup>σ<sub>1</sub></sup> · ζ<sub>1</sub><sup>δ</sup>)<sup>1/γ</sup> ⇒ β<sub>1</sub><sup>1/γ</sup> = g<sup>σ<sub>1</sub>/γ</sup> · g<sup>rnd\*·δ</sup> ⇒ B<sub>1</sub> = g<sup>σ'<sub>1</sub></sup> · z<sub>1</sub><sup>δ'</sup>.
- Per the validity of the signature, we have  $\beta_2 = h^{\sigma_2} \cdot \zeta_2^{\delta} \Rightarrow \beta_2^{1/\gamma} = (h^{\sigma_2} \cdot \zeta_2^{\delta})^{1/\gamma} \Rightarrow \beta_2^{1/\gamma} = h^{\sigma_2/\gamma} \cdot (z/g^{\text{rnd}^*})^{\delta} \Rightarrow \beta_2^{1/\gamma} = h^{\sigma_2/\gamma} \cdot (z/z_1)^{\delta} \Rightarrow B_2 = h^{\sigma_2'} \cdot z_2^{\delta'}.$

# **D DEFERRED FIGURES**

#### Secure Showing of Partial Attributes

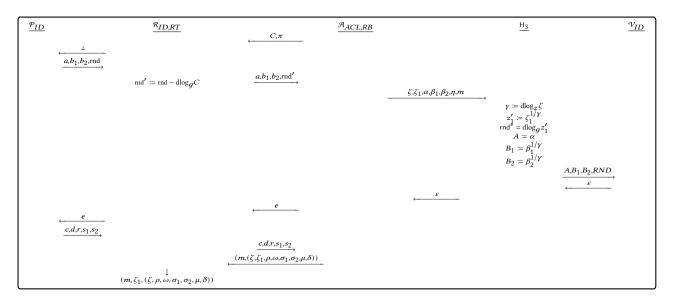


Figure 2: A reduction from RB of ACL to the RT of ID<sub>ACL</sub>.

$S(sk = x, pk = (h, y = g^x, z), (h_0, \dots, h_n))$		$\mathcal{U}(m,pk = (h, y, z), (L_1, \dots, L_n), (h_0, \dots, h_n))$
<b>Registration</b> <b>Signing</b> $d, s_1, s_2, u, \text{rnd} \leftarrow_{\$} \mathbb{Z}_p$ $z_1 := g^{\text{rnd}} \cdot C, z_2 := z/z_1$ $a := g^{\mu}$	<u>С,</u> π	$r_{com} \leftarrow \mathbb{S} \mathbb{Z}_p$ $C := h_0^{r_{com}} \cdot h_1^{L_1} \cdot \ldots \cdot h_n^{L_n}$ $\pi$
$b_1 \coloneqq g^{s_1} \cdot z_1^d, b_2 \coloneqq h^{s_2} \cdot z_2^d$	$\xrightarrow{a,b_1,b_2,\mathrm{rnd}}$	If rnd = 0 or $a \notin \mathbb{G} \lor b_1 \notin \mathbb{G} \lor b_2 \notin \mathbb{G}$ : output $\perp$ $z_1 = g^{\text{rnd}} \cdot C, \gamma \leftarrow_S \mathbb{Z}_p^*$ $\zeta = z^{\gamma}, \zeta_1 = z_1^{\gamma}, \zeta_2 = \zeta/\zeta_1$ $\tau, t_1, t_2, t_3, t_4, t_5 \leftarrow_S \mathbb{Z}_p$ $a := a \cdot g^{t_1} y^{t_2}, \eta := z^{\tau}$ $\beta_1 := b_1^{\gamma} \cdot g^{t_3} \cdot \zeta_1^{t_4}, \beta_2 := b_2^{\gamma} \cdot h^{t_5} \cdot \zeta_2^{t_4}$ $\varepsilon := H_3(\zeta, \zeta_1, \alpha, \beta_1, \beta_2, \eta, m)$
$c := e - d \mod q$ $r := u - c \cdot x \mod q$	cdrs. so	$\begin{split} e &= \varepsilon - t_2 - t_4 \\ \rho &\coloneqq r + t_1, \omega \coloneqq c + t_2 \\ \sigma_1 &\coloneqq s_1 \cdot \gamma + t_3, \sigma_2 \coloneqq s_2 \cdot \gamma + t_5 \\ \delta &\coloneqq d + t_4, \mu \coloneqq \tau - \delta \cdot \gamma \\ \text{If } \omega + \delta &= \text{H}_3(\zeta, \zeta_1, g^{\rho} \cdot y^{\omega}, g^{\sigma_1} \cdot \zeta_1^{\delta}, h^{\sigma_2} \cdot \zeta_2^{\delta}, z^{\mu} \cdot \zeta^{\delta}, m): \\ \text{Output the Signature } (\zeta_1, (\zeta, \rho, \omega, \sigma_1, \sigma_2, \mu, \delta)) \\ \text{Else: output } \bot \end{split}$

Figure 3: The ACL scheme [3] depicted as an interactive protocol. The check  $\omega + \delta = H_3(\zeta, \zeta_1, g^{\rho} \cdot y^{\omega}, g^{\sigma_1} \cdot \zeta_1^{\delta}, h^{\sigma_2} \cdot \zeta_2^{\delta}, z^{\mu} \cdot \zeta^{\delta}, m)$  also constitutes the verification equation for a resulting signature.