Anonymous Credentials Light

Foteini Baldimtsi, Anna Lysyanskaya
foteini,anna@cs.brown.edu

Computer Science Department, Brown University

Abstract. We define and propose an efficient and provably secure construction of blind signatures with attributes. Prior notions of blind signatures did not yield themselves to the construction of anonymous credential systems, not even if we drop the unlinkability requirement of anonymous credentials. Our new notion in contrast is a convenient building block for anonymous credential systems. The construction we propose is efficient: it requires just a few exponentiations in a prime-order group in which the decisional Diffie-Hellman problem is hard. Thus, for the first time, we give a provably secure construction of anonymous credentials that can work in the elliptic group setting without bilinear pairings and is based on the DDH assumption. In contrast, prior provably secure constructions were based on the RSA group or on groups with pairings, which made them prohibitively inefficient for mobile devices, RFIDs and smartcards. The only prior efficient construction that could work in such elliptic curve groups, due to Brands, does not have a proof of security.

Keywords: Anonymous credentials, attributes, blind signatures, lightweight devices, private identity management.

1 Introduction

Anonymous credentials, envisioned by David Chaum [16], and first fully realized by Camenisch and Lysyanskaya [12], have proved to be a centrally important building block in privacy-minded identity management systems currently under development [42, 35, 15]. They allow users to prove possession of credentials without revealing any other information about themselves; when such a proof is carried out, it cannot be linked to previous uses of the same credential, or to any other identifying information about the user. Additionally, they give the users the ability to privately obtain credentials. The reason that they have become so popular is that they strictly adhere to data minimization principles [31]: no electronic transaction should require its participants to needlessly reveal private information. These principles are at the core of European privacy standards [37, 36] and also the National Strategy for Trusted Identities in Cyberspace (NSTIC) published by the U.S. government [18].

Efficiency considerations are important when deploying these algorithms in practice. While on a PC, the Camenisch-Lysyanskaya credentials [12] and follow-up work [4, 9, 10, 13] take only a fraction of a second of CPU time, in mobile devices, smartcards and RFID cards, they are not yet as practical as we may wish for certain applications. The issue is that all existing anonymous credential systems use either the RSA group, or groups that admit bilinear pairings, and the security parameter choice needed to provide security under the required assumptions in these groups makes these systems too expensive for smartcards and mobile devices (i.e. original CL credentials require an RSA group with bitlength 2048 or a pairing with 128 bits of security). Bichsel et. al. [5] show an efficient implementation of CL credentials on a standard Java card, where they achieve an execution time for an RSA exponentiation for a base of 1984 bits and an exponent of 1024 bits of 1.3 seconds by using the hardware accelerator for RSA of the chosen card. As a result, more than 16 seconds are required in order to show one credential which is not practical for most applications. Moreover, hardware implementations have high development cost (compared to software) and are not easy to update.

From the efficiency point of view, therefore, the UProve credential system [35] (based on Brands' work [7, 8], acquired and implemented by Microsoft) seems attractive. UProve does not allow unlinkable reuse of credentials: in order to unlinkably use a credential again, a user must get it reissued. Yet, these "linkable" credentials can serve as the basis for e-cash, where a user is not supposed to use a coin more than once, anyway. The UProve credential issuing protocol, as well as the algorithm for verifying a credential, are practical enough for the more advanced end of RFID technology, since they can work in any group where the discrete logarithm problem is hard (thus can be based on ECC), and require just a few exponentiations in this group. Software implementations of ECC multiplication are much faster than the equivalent RSA exponentiation [28] (hardware accelerators could also speed up both ECC and RSA implementations by an order of magnitude or more; today there exist RFID chips that perform ECC multiplication in 0.4 seconds [43]).

So perhaps one could use these "linkable" anonymous credentials and just have them reissued as many times as needed. However, UProve is not provably secure! No proof of security has been given under any reasonable assumption; and in fact, recently, Baldimtsi and Lysyanskaya [3] showed that all known approaches for proving security in the random oracle model will fail for the Brands blind signature scheme on which UProve is based, under essentially any assumption.

One might ask what happens if, instead of the Brands blind signature, one tries to construct linkable anonymous credentials from a provably secure blind signature scheme. Unfortunately, blind signatures as traditionally defined [40, 39, 30, 1] do not give any consideration to users' attributes. In a blind signature, a user is free to choose whichever message he wants signed. In a credential scheme, a user should be able to get his particular attribute set signed by the signer. The signers can verify that what they are signing is the correct attribute set, even though they cannot see exactly what they are signing. For example, a signer might want to sign a user's secret key, his age, and his citizenship information, and wants to be able to verify that this is indeed what she is signing. Attributes are, in fact, essential: without them, it is impossible to tie a credential to a particular user's identity, and so users would be able to just pool each others' credentials as desired, instead of tying them to a particular secret key. Moreover, giving the user the ability to choose any message means that the content of the signed messages has not been vetted by the signer at all.

Thus, a better question is: how do we extend the notion of a blind signature so that it can efficiently accommodate anonymous credentials — if not full anonymous credentials a-la Camenisch and Lysyanskaya, then at least linkable lightweight anonymous credentials a-la Brands?

Blind Signature with Attributes. In this paper, for the first time, we define *blind signatures with attributes*. In such a blind signature scheme, the signer and the user both get as input a cryptographic commitment C to the user's attributes; this way, the user can prove that the commitment contains the correct attributes via a separate zero-knowledge proof. As output, the user obtains another, unlinkable, commitment $\tilde{C}$ to the same attributes, and a signature on this commitment and a message of the user's choice. *Blindness* ensures that, even upon seeing two signatures obtained this way on commitments of his own choice, the signer cannot link a signature to its issuing. *Unforgeability* ensures that a user cannot produce more signatures than he was issued, and also that the multiset of openings to the input commitments is the same as the multiset of openings to the output commitments.

From Blind Signatures with Attributes to Anonymous Credentials. It is easy to see that this is the right building block for linkable anonymous credentials: a user with a particular set of attributes can form a commitment C to these attributes, prove in zero-knowledge that he has committed to the correct attributes, and then obtain a credential by running a blind signature with attributes on input this commitment. He can then prove that he has a credential with the desired attributes by revealing his signature and the output commitment $\tilde{C}$, and proving in zero knowledge that $\tilde{C}$ corresponds to the needed attributes.

For example, suppose that we allow users to obtain and anonymously show age credentials. Then Alice will form a commitment C to her secret key sk and her age age, prove to the signer, who serves as an identity validator, that she has committed to sk that corresponds to her pk and to the correct age, and run a blind signature with attributes protocol to obtain a fresh commitment $\tilde{C}$ on the same sk and age, and a signature σ. Then, when she needs to prove her age, she can reveal $(\tilde{C}, \sigma)$ and prove that the age inside commitment $\tilde{C}$ allows her entry to some age-restricted resource, for example a video store that requires viewers to be over 13. If she wants to do it again, she needs to run the blind signature with attributes protocol with the signer again. She can further anonymously obtain credentials that are connected to her identity: let's say that the video store wants to give her a credential for a free movie; she can obtain it by running a blind signature with attributes with the video store. She will form her input commitment C' by committing to the same sk and age as in $\tilde{C}$ and proving to the video store that she did so (by proving equality of committed values); once she runs the protocol with the video store, she receives an output $(\tilde{C'}, \sigma)$, which is a credential from the video store (and not from the original signer) on her (sk, age) — even though the video store never saw the public key at all. More interestingly, we can require the free movie to be a single-use credential, by additional clever use of attributes (see Section 4), so that Alice can be traced if she tries to get two free movies using the same single-use credential more than once. Thus we see that blind signatures with attributes are the right building block for linkable anonymous credentials.

Construction. In addition to the definition, we solve the open problem of constructing anonymous credentials based on the DDH assumption. We give a lightweight construction of blind signatures with attributes; because of the connection to single use anonymous credentials we call our construction "anonymous credentials light" (ACL). In the random-oracle model, our construction is unlinkable under the decisional Diffie-Hellman assumption, and unforgeable under the discrete-logarithm assumption for sequential composition (we leave extending it to concurrent self-composition as an open problem). Our construction is inspired by the Abe blind signature [1] in which, unlike other provably secure blind signatures [6, 11, 34, 40, 22], blinding still preserves some structural elements into which attributes can be embedded. Unfortunately Abe's construction, as it was first presented, had a faulty proof of security as it was later found by Abe and Ohkubo [33]. The original proof was in the RO model for concurrent composition but it turned out to only be valid for an adversary with overwhelming success probability so they ended up presenting a new proof in the generic model. In the other hand, we manage to proof our scheme in the RO model by limiting our attention to sequential composition only. Although not shown, one could adjust our technique to also prove Abe's blind signature secure in the RO model under sequential composition.

Related Work and Comparisons. A related work is due to Guajardo et al. [26] where they proposed "encrypted anonymous credentials". The idea is that an issuer can certify encrypted attributes in such a way that none of the involved parties, including the user, can learn the value of the attributes. This is quite different from our proposal of blind signatures with attributes as a general building block. We consider users who know the values of their own attributes and use them to form commitments that the issuer signs. Our scheme can easily be adapted to capture the case where the issuer may learn a a subset of the attributes he is signing but still the user knows his attributes. Moreover, their construction [26] is based on Brands scheme which as mentioned above cannot proven one-more unforgeable [3].

Let's now take a closer look in our ACL construction and compare it with other well known blind signature and credential schemes. For our comparisons we focus on the efficiency of signing (which is analogous to the credential issuing), the cost of verification of the signature/credential, the size of the signature/credential, the security of the schemes i.e. blindness and unforgeability and whether the schemes supports attributes which is obviously not going to be the case for a "traditional" blind signature scheme.

The ACL construction consists of three phases: registration, preparation and validation. Registration is the most expensive phase, but it need not be repeated for the same user/signer pair should the user need his credential reissued (as long as his attribute set is the same). Preparation and validation require the signer to perform 7 exponentiations, while the user performs 13; verification requires 8 exponentiations. These numbers are essentially the same as Abe's blind signature; Brands' blind signature at the heart of UProve requires essentially the same number of exponentiations for signature issue, and 4 exponentiations fewer for verification. Thus, we get comparable efficiency to the most efficient examples of protocols that either lack essential features (such as previously provable secure blind signatures) or provable security (such as UProve). We also compare ourselves to the Camenisch-Lysyanskaya [12] which is significantly less efficient during issuing due to the use of pairings or RSA group but is a multi-use credential, thus we denote the signature/credential size to be close to zero since it can be used multiple times. See Table 1 for a comparison of the above mentioned schemes.

	Brands [7]		**Abe [1]**		**ACL Scheme**		**CL credentials [?]**			
							Rsa		**Pairings**	
Efficiency	S[1]	U[2]	S	U	S	U	S	U	S	U
Signing[3]	2	12	6	12	7	13	10	8	15	14p
Verification	7	0	11	1	11	1	NC[4]		5+6p	9+6p
Sign. size	6 elem.		9 elem.		9 elem.		~ 0[5]		~ 0[5]	
Blindness	✓		✓		✓		✓		✓	
Prov. Unforg.	✗		GM[6]		✓		✓		✓	
Attributes	✓		✗		✓		✓		✓	

[1] Signer, [2] User, [3] In number of exponentiations
[4] Non-comparable, [5] It can be used multiple times
[6] Generic group model

Table 1. Comparison of anonymous credentials schemes

Implementation. In an independent to this work [29] our ACL scheme has been implemented using an NFC smartphone: the BlackBerry Bold 9900. The implementation has been tailored for payments in transportation systems where our ACL construction is being used as an e-cash scheme (which as we mentioned above is a direct application of "single-use" credentials). The implementation has been based on eliptic curves and the results are very promising. Briefly, the signing/issuing takes a total of 300 milliseconds including terminal, communication and smartphone execution time, while spending (i.e. verification of the signature) takes about 380 milliseconds when 2 attributes are revealed which would be less if not attributes where revealed.

2 Preliminaries

2.1 Witness Relations and Interactive Proofs

A witness relation for a language $L \in \mathcal{NP}$ is defined as:

Definition 1. *(Witness relation [23]) A witness relation for a language $L \in \mathcal{NP}$ is a binary relation R_L that is polynomially bounded (i.e., $(h, x) \in R_L$ implies $|x| \leq poly(|h|)$), is polynomial-time-recognizable and characterizes L by*

$$L = \{h : \exists x \ s.t. \ (h, x) \in R_L\}.$$

For $h \in L$, any x satisfying $(h, x) \in R_L$ is called a *witness* (for the membership $h \in L$). By $R_L(h)$ we denote the set of witnesses for the membership $h \in L$; that is, $R_L(h) = \{x : (h, x) \in R_L\}$.

A relation R_L is said to be "hard", if one can efficiently generate (h, x)'s such that, when given h it is hard to find the witness x.

Definition 2. *(Interactive Proof System [24]) An interactive proof system with soundness error $s \in [0, 1]$ for a language L with witness relation R_L is a pair of algorithms (P, V) where v is probabilistic polynomial time and the following properties hold:*

[noitemsep]

1. *Completeness. For every $h \in L$ and every $x \in R_L(h)$,*

$$Pr[\langle P(x), V_L \rangle(h) = 1] = 1.$$

2. s-Soundness. For every $h \notin L$, every $z \in \{0,1\}^*$ and every interactive algorithm P^*

$$Pr[\langle P^*(z), V\rangle(h) = 0] \geq 1 - s.$$

A useful property in the above setting would be if the verifier V wouldn't learn anything useful from P about the witness x besides the fact that P knows x. This property is called *zero knowledge*. If soundness error s is negligible, then this interactive proof system has *strong soundness*.

Definition 3. *(Honest Verifiable Zero Knowledge (HVZK)) An interactive proof system (P, V) for a language L is said to be* honest verifiable zero knowledge *if there exists a probabilistic polynomial time algorithm S (the Simulator) such that for all $h \in L$:*

$$view_V[P(h) \leftrightarrow V(h)] \approx S(h),$$

where view_V is the view of the honest verifier V of the interaction between V and P on input h.

2.2 Σ-Protocols

Σ-protocols are a class of interactive proofs where the Prover and the Verifier (P, V) have a common input h and P proves in zero-knowledge that he knows a value x such that $(h, x) \in R_L$. Their main characteristic is that they have exactly 3 rounds of the following type: (1) P sends a message a to V, (2) V responds with a random challenge c chosen from a domain of size $\Theta(k)$ and (3) P resents a reply r. V decides whether to accept or not given the information he has seen: (h, a, c, r). Formally:

Definition 4. *(Σ-Protocol) A protocol $\mathcal{P}$ is said to be a Σ-protocol for a relation R_L if:*
[noitemsep]
- *$\mathcal{P}$ is of the above three rounds form, and if (P, V) follow the protocol, the verifier always accepts.*
- *From any h and any pair of accepting conversations on input h, (a, c, r), (a, c', r') where $c \neq c'$, one can efficiently compute x such that $(h, x) \in R_L$ (special soundness).*
- *There exists a polynomial-time simulator S, which on input h and a random c outputs an accepting conversation of the form (a, c, r), with the same probability distribution as conversations between the honest P, V on input h (special honest-verifier zero-knowledge).*

An example of a Σ-protocol is the Schnorr Identification scheme [41]. Let G be a group of prime order q with generator g, and let $\mathbb{Z}_q$ denote the field of integers modulo q. Schnorr's identification scheme works as follows:

$$\begin{array}{lcl}
\text{Prover}(q, g, h = g^x) & & \text{Verifier}(q, g, h) \\
\hline
y \leftarrow \mathbb{Z}_q,\ a = g^y & \xrightarrow{\ a\ } & \\
 & \xleftarrow{\ c\ } & c \leftarrow \mathbb{Z}_q \\
r = y + cx \bmod q & \xrightarrow{\ r\ } & g^r \overset{?}{=} ah^c
\end{array}$$

Σ-protocols are an essential building block for blind signatures and anonymous credentials. For example Brands' [7] and Abe's [1] schemes are based on a Σ-protocol, while CL anonymous credentials [12] uses ZK proofs which are based on Σ-protocols.

2.3 Fiat-Shamir Heuristic

Fiat and Shamir [21] proposed a method to transform any three-round interactive proof system with negligible soundness error, such as Σ-protocols, into a digital signature scheme using a hash function, modeled as a random oracle.

To transform a three-round proof system into a signature scheme, one could, instead of a random c, compute $c = H(a, m)$, where $H \to \{0,1\}^*$ is a hash function. Then, the Fiat-Shamir transformation uses c to create a signature $\sigma(m) = (a, r)$. Specifically:

– $\mathrm{Gen}(1^k) : (h, x) \leftarrow R_L, \mathrm{SK} = x, \mathrm{PK} = (h, H)$.
– $\mathrm{Sig}(m)$: produce a honestly using x, set $c = H(m, a)$, produce r honestly for this c, output $\sigma(m) = (a, r)$.
– $\mathrm{Ver}(m, (a, r))$: $c = H(a, m)$ and check if (a, c, r) is valid.

Famous digital signatures that have been constructed from Σ-protocols using the Fiat-Shamir heuristic include Schnorr's [41] and GQ signatures [27] and they have been proven secure in the RO model [39]. Note that, it has also been shown that the digital signatures produced by the Fiat-Shamir heuristic are not secure in the standard model [25].

2.4 OR-proof

Let $\mathcal{P}$ be a Σ-protocol for a relation R_L, (h_0, h_1) be a common input to (P, V) and P knows a x such that $h_b, x \in R_L$ for $b \in \{0, 1\}$. An OR-proof protocol $\mathcal{P}_{OR}$ is a Σ-protocol for proving that either $(h_0, x) \in R_L$ or $(h_1, x) \in R_L$ [20][1].

The main idea of an OR-proof is that P will complete two Σ protocols Σ_0, Σ_1, one for h_0 and one for h_1 in such a way that the verifier will not be able to tell for which of the two P knows the corresponding witness. For h_b the prover can actually follow the real Σ protocol while for h_{1-b} he will have to use a simulator M to create his answers. A $\mathcal{P}_{OR}$ protocol works as follows:

1. P computes the first message of Σ_b, a_b, using (h_b, x) as input. P randomly chooses c_{1-b} and runs the simulator M on input (h_{1-b}, c_{1-b}) and receives $(a_{1-b}, c_{1-b}, r_{1-b})$ as an output. Then, P sends a_0, a_1 to V.
2. V chooses a random string e and sends it to P.
3. P sets $c_b = e \oplus c_{b-1}$ and computes the answer r_b to challenge c_b using h_b, a_b, c_b, x as input. He sends c_0, c_1, r_0, r_1 to V.
4. V checks that $e = c_0 \oplus c_1$ and that both (a_0, c_0, r_0) and (a_1, c_1, r_1) are accepting conversations.

Let $R_{OR} = \{((h_0, h_1), x) | (h_0, x) \in R_L \textit{ or } (h_1, x) \in R_L\}$. Then:

Theorem 1 ([20]). *The $\mathcal{P}_{OR}$ protocol is a Σ-protocol for R_{OR}. Moreover, for any verifier V^*, the probability distribution of conversations between P and V^*, where x is such that $(h_b, x) \in R_L$, is independent of b.*

2.5 Generalized Pedersen Commitment

A non-interactive commitment takes as input a message (or set of messages) m and randomness R and outputs a value that, on the one hand, reveals no information about the message but, on the other hand, it is hard to find a (m', R') such that $\mathit{Commit}(m; R) = \mathit{Commit}(m', R')$ but $m \neq m'$ (see [32] for a standard definition and treatment). The Pedersen commitment is a commitment scheme based on the discrete logarithm assumption [38]. Here we give a generalized version that allows a commitment to a set of messages $(L_1, \ldots, L_n)$. The scheme is defined as follows:

1. *Setup*: On input the security parameter 1^k and the maximum number of messages n, pick a group G of prime order $q = \Theta(2^k)$ with generators $h, h_1, \ldots, h_n$.
2. $\mathit{Commit}(L_1, \ldots, L_n; R) = h^R \prod_{i=1}^{n} h_i^{L_i}$; note that $L_i \in \mathbb{Z}_q$.

The Pedersen commitment scheme (constructed from a corresponding Σ-protocol) is information theoretically hiding and computationally binding. It is also important to note that there are efficient zero-knowledge proof protocols for proving that a commitment C is to a particular set of values; or to a set of values that satisfy a rich class of relations [7, 14, 19].

[1] This is based on a more general result due to Cramer et. al. [17] where they present a protocol in which the prover demonstrates knowledge of the solution to some subset of n problem instances out of a collection of subsets.

2.6 Combined Commitment Scheme

Let Commit_1 be a commitment scheme that takes as input its parameters params_1, n messages $(L_1, \ldots, L_n)$ to which we will refer to as *attributes* from now on and randomness R_1 and outputs a commitment C_1. Let Commit_2 be a commitment scheme that takes as input its parameters params_2, an attribute L_0 and randomness R_2 and outputs a commitment C_2. Suppose that another commitment scheme, *Commit*, is a *combination* of these two commitments; i.e., on input C_1 and C_2 it produces a commitment C to the combined attributes $(L_0, L_1, \ldots, L_n)$ with combined randomness $R = R_1 + R_2$.

For example, this can be instantiated by a generalized Pedersen commitment scheme: the parameters for the combined scheme are generators $(h, h_0, \ldots, h_n)$, $\mathit{Commit}_1(L_1, \ldots, L_n; R_1) = (\prod_{i=1}^{n} h_i^{L_i})h^{R_1}$, $\mathit{Commit}_2(L_0; R_2) = h_0^{L_0}h^{R_2}$, and a combined commitment can be obtained either by multiplying together the two component commitments, or by computing it from scratch as $\mathit{Commit}(L_0, L_1, \ldots, L_n; R) = (\prod_{i=0}^{n} h_i^{L_i})h^R$.

2.7 Blinded Pedersen Commitment Scheme

Here, we note that the Pedersen commitment scheme can be further extended. Let $(h, h_1, \ldots, h_n)$ be the parameters of the Pedersen commitment. Consider an additional parameter $z \in G$, where $z \neq 1$. Let $C = \mathit{Commit}(L_1, \ldots, L_n; R)$. Then the values (z^γ, C^γ) can also be viewed as a commitment to the same $(L_1, \ldots, L_n)$ with randomness (R, γ). Let us define a new commitment scheme, which we will call the *blinded* Pedersen commitment scheme: $\mathit{Commit}^B(L_1, \ldots, L_n; R, \gamma) = (z^\gamma, \mathit{Commit}(L_1, \ldots, L_n; R)^\gamma)$, where *Commit* is the Pedersen commitment. It is easy to see that this commitment is unconditionally hiding, same as Pedersen. It is also easy to see that it is binding: given (z^γ, C^γ), γ is uniquely defined, and therefore so is C, which is binding. Finally, using well-known Σ-protocols, it is easy to see that the same set of relations that can be proven about values inside a Pedersen commitment can be proven about values inside a blinded Pedersen commitment.

3 Definition of Secure Blind Signatures with Attributes

In this section, for the first time, we define what a blind signature scheme with attributes is and its security properties. For definitions of standard (without attributes) blind signatures please refer to [30].

Definition 5. *(Blind Signatures with attributes)*
Let Commit$(x; r)$ be a non-interactive commitment scheme whose parameters params are generated by a trusted party TP and let n be the maximum number of messages the scheme can support . A blind signature scheme with n attributes, for this commitment scheme, consists of three algorithms: KeyGen, BlindSign, Verify where BlindSign is a protocol between the Signer(S) and the User(U).

- *KeyGen$(1^k, \mathit{params})$: is a probabilistic polynomial time key generation algorithm. It takes as input the security parameter k and the public parameters of the commitment scheme and outputs (pk, sk) where pk is the signer's public key and sk is his secret key.*
- *BlindSign: is an interactive, probabilistic polynomial time protocol between S and U. The public parameters are the Signer's public key pk, the parameters of the commitment scheme params and $C = \mathit{Commit}(L_1, \ldots, L_n; R)$ where $(L_1, \ldots, L_n)$ is the set of attributes and R is some randomness. The Signer's private input is sk and User's private input is $(L_1, \ldots, L_n, R)$ and the message, m, that he wishes to have signed. The User's output in the protocol is a pair $(\tilde{R}, \sigma)$, where $\sigma = \sigma(m, \tilde{C})$ is the Signer's signature on $(m, \tilde{C})$, and $\tilde{C} = \mathit{Commit}(L_1, \ldots, L_n; \tilde{R})$; the Signer's output is "completed".*
- *Verify$(PK, m, \tilde{C}, \sigma)$: is the signature verification algorithm; i.e. a deterministic polynomial time algorithm that gets as input the public key, the message, a commitment to the attributes and the blind signature on the message and attributes and checks the validity of the signature σ. If it is valid the algorithm outputs "1", otherwise outputs "0".*

A blind signature scheme with attributes is secure if it is both blind and unforgeable. Blindness is defined in a similar way as in blind signature schemes without attributes: the Signer is unable to view the messages and the attributes he signs (protection for the User). A malicious Signer, $\mathcal{A}$, cannot link a $(m, \tilde{C}, \sigma)$ tuple to any particular execution of the protocol, even if $\mathcal{A}$ chooses $m, L_1, \ldots, L_n$.

Definition 6. *(Blindness for blind signatures with attributes) Let $\mathcal{A}$ be a malicious Signer and $b \in \{0,1\}$ be a randomly chosen bit which is kept secret from $\mathcal{A}$. $\mathcal{A}$ will try to guess the value b by performing the following steps:*
[noitemsep]

1. $(pk, sk) \leftarrow KeyGen(1^k)$
2. $\{m_0, m_1, \overrightarrow{L_0}, \overrightarrow{L_1}, R_0, R_1\} \leftarrow \mathcal{A}(1^k, pk, sk)$ *(i.e. $\mathcal{A}$ produces two messages $\{m_0, m_1\}$, polynomial in 1^k, and two attribute vectors $\overrightarrow{L_0}, \overrightarrow{L_1}$ with the corresponding randomness).*
3. $\mathcal{A}(1^k, pk, sk, m_0, m_1, \overrightarrow{L_0}, \overrightarrow{L_1}, R_0, R_1)$ *engages in two parallel (and arbitrarily interleaved as desired by $\mathcal{A}$) interactive protocols, the first with $U(pk, \{m_b, \overrightarrow{L_0}, R_0\})$ and the second with $U(pk, \{m_{1-b}, \overrightarrow{L_1}, R_1\})$.*
4. *$\mathcal{A}$ only gets signatures and the corresponding blinded commitments for the instances that didn't fail; for the ones that failed $\mathcal{A}$ gets nothing. If neither of the User instances failed, then $\mathcal{A}$ gets two signatures and the corresponding blinded commitments: $\sigma(m_0, \tilde{C}_b), \tilde{C}_b$ and $\sigma(m_1, \tilde{C}_{1-b}), \tilde{C}_{1-b}$.*
5. *$\mathcal{A}$ outputs a bit b'.*

Then the probability, taken over the choice of b, over coin-flips of the key-generation algorithm, the coin-flips of $\mathcal{A}$, and (private) coin-flips of both users (from step 3), that $b' = b$ is at most $\frac{1}{2} + \nu(k)$, where $\nu(k)$ is a negligible function.

We give the definition of one-more unforgeability for the sequential composition case.

Definition 7. *(Sequential one-more unforgeability for blind signatures with attributes) (KeyGen, BlindSign, Verify) is a one-more unforgeable blind signature scheme with respect to Commit if $\forall$ ppt $\mathcal{A}$, the probability that $\mathcal{A}$ wins in the following game is negligible:*
[noitemsep]

1. $(pk, sk) \leftarrow KeyGen(1^k)$
2. *$\mathcal{A}(pk, params)$ engages in polynomially many (in k) adaptive, sequential interactive protocols with polynomially many copies of the signer, where $\mathcal{A}$ decides in an adaptive fashion when to stop. For every execution, $\mathcal{A}$ forms a commitment C_i and picks a message m_i and invokes $S(pk, sk, C_i)$. Let ℓ be the number of executions, where the Signer output "completed" in the end of the protocol.*
3. *$\mathcal{A}$ outputs a collection $\{(\tilde{C}_1, m_1, \sigma_1), \ldots, (\tilde{C}_j, m_j, \sigma_j)\}$ where $(\tilde{C}_i, m_i, \sigma_i)$ for $1 \leq i \leq j$ are all accepted by $Verify(pk, \tilde{C}_i, m_i, \sigma_i)$, and all $(\tilde{C}_i, m_i)$'s are distinct.*

We say that $\mathcal{A}$ wins the game if either:

1. *$j > \ell$ (i.e. $\mathcal{A}$ outputs more $(\tilde{C}, m, \sigma)$ tuples than he received).*
2. *$\mathcal{A}$ opens the sets of commitments $\{C_i\}$ and $\{\tilde{C}_i\}$ and the resulting multisets do not match.*

4 From Blind Signatures with Attributes to Single-Use Credentials

Single-use anonymous credentials ensure that a credential cannot be used more than once: if used more than once, the user's identity can be discovered, and the user can be penalized. This is a generalization of off-line electronic cash [16], in that such a credential can also contain attributes, while e-cash does not necessarily have any connection to a user's identity. Here we construct single-use anonymous credentials from blind signatures with attributes using standard techniques [16, 7, 10].

Recall our definition of combined commitment schemes ($Commit_1, Commit_2, Commit$) from Section 2.6. Consider the following construction of a single-use credential with n attributes from a blind signature scheme with $n+1$ attributes. The common inputs to the User and Signer are the Signer's public key for verifying signatures, and a commitment C_1 to the User's attributes $(L_1, \ldots, L_n)$. We assume that the attribute L_1 contains the user's identity, so that learning that attribute will allow one to trace the user. The Signer's private input is its signing key, while the User's private input is the message m that the user wants signed (m will serve as the serial number for this credential, so the user needs to have chosen it at random during the signing protocol) and the opening to C_1, $(L_1, \ldots, L_n; R_1)$. The commitment C_1 corresponds to this user's identity and never changes; each time this user obtains a signature, they input the same C_1.

First, the User forms a commitment $C_2 = Commit_2(L_0; R_2)$ for random (L_0, R_2); this commitment is specific to this credential, and the value L_0 will be used in the double-spending equation, below. Next, the user submits C_2 to the signer and provides a zero-knowledge proof of knowledge of its opening.

Next, the User and the Signer carry out the blind signature protocol relative to the combined commitment scheme *Commit*, whereby the User's output is a fresh commitment $\tilde{C}$ to the values $(L_0, \ldots, L_n)$ and randomness $\tilde{R}$, and the signature σ on $(m, \tilde{C})$.

To use this credential to a verifier V, the User reveals $(m, \tilde{C}, \sigma)$, obtains from V a challenge c, and reveals the double-spending equation $cL_1 + L_0$ (the multiplication and addition is in whatever ring the committed values belong to) and whatever other attributes are needed for the transaction; the User then provides a zero-knowledge proof that these values were revealed correctly, relative to commitment $\tilde{C}$.

To trace a user who spent a single-use credential twice, it is sufficient to examine two double-spending equations for the same credential. If double spending occurred, then the attribute L_1, which encodes the User's identity, will be revealed.

The following theorem is stated informally because we omitted a formal definition of single-use anonymous credentials.

Theorem 2. *(Informal) Let KeyGen, BlindSign, Verify be a blind signature scheme with $n+1$ attributes, associated with the commitment scheme Commit which is a combination of $Commit_1$ and $Commit_2$ as explained above. Then the above construction is a single-use credential scheme with n attributes.*

5 Our Construction: ACL

Let's now describe our proposed construction of a blind signature scheme with attributes, called "Anonymous Credentials Light" (ACL). ACL is based on Abe's blind signature scheme [1] which has been modified appropriately so that it will also allow the users to encode attributes in the signatures they get in a provably secure way. Abe's scheme is a witness indistinguishable variant of the Schnorr signature scheme where the Signer owns a *real public key* $y = g^x$ and a *tag public key* $z = g^w$. The idea is that the signature can be issued only with the real secret key x but no one can distinguish which of the two (real or tag) secret keys was used (remember the OR-proof technique presented above).

Although the modification seems straightforward, proving the security of the resulting scheme turns out to be challenging as we will see later in the next section.

5.1 Setup

Let G be a group of order q and g is a generator of this group. Let $\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_q$ be a hash function. We assume the existence of a trusted party, TP, which chooses G, g and outputs $params = (q, G, g, z, h, h_0, \ldots, h_n)$ where $z, h, h_0, \ldots, h_n \in G$ and n is the maximum number of attributes a User can possibly have. We will use $(h_0, \ldots, h_n)$ as parameters of the combined Pedersen commitment scheme *Commit*; we will use $(z, h_0, \dots, h_n)$ as parameters for the blinded Pedersen commitment scheme Commit^B.

The Signer picks his secret key $x \in_R \mathbb{Z}_q$ and computes his *real public key* to be $y = g^x \bmod q$. The TP given (G, q, g, h, y) outputs z which is the *tag public key* of the Signer. The public key of the Signer is (G, q, g, h, y, z), and the private key is x.

5.2 Signature/ Credential Issuing

The signature issuing protocol will be described in three phases: registration, preparation and validation. The registration phase actually only needs to happen once for each user/set of attributes and also, preparation and validation phases can happen simultaneously (we choose to present them separately in order to describe the construction in a more modular way).

The signature issuing protocol is basically a $\mathcal{P}_{OR}$-protocol for proving knowledge of one of the following:

- y-side: proof of knowledge x of $y = g^x$
- z-side: proof of knowledge (w_1, w_2) of $z_1 = g^{w_1},\ z_2 = h^{w_2}$ (where z_1, z_2 are the so called "one-time" tag keys that the signer creates).

Notice that the z-side witness in not known to the Signer, so the z-side proof will be done by simulation following the OR-proof technique paradigm described in Appendix 2.4.

Registration The User's input includes the system parameters *params*, the signer's real public key y, the message m to be signed and $(L_1, \dots, L_n; R)$ where $L_1, \dots, L_n$ is a set of attributes and R is some randomness. The Signer also gets as input the system parameters *params*, a commitment $C = \mathit{Commit}(L_1, \dots, L_n; R)$ and his secret key x. During registration the User and the Signer carry out a standard interactive zero-knowledge proof of knowledge protocol where the User creates a proof π_1 to convince the Signer that he knows an opening of the commitment C.[2]

Preparation The Signer prepares z_1 and z_2 (the z-side proof). He first picks $\mathit{rnd} \in \mathbb{Z}_q$ and creates the "one-time" tag keys: $z_1 = Cg^{\mathit{rnd}}$ and $z_2 = z/z_1$. The Signer sends rnd to the User in order to convince him that $\log_g z_1$ is not known to him. The User first checks that $\mathit{rnd} \neq 0$ and then computes himself $z_1 = Cg^{\mathit{rnd}}$ and then picks $\gamma \in \mathbb{Z}_q^*$ and blinds z, z_1, z_2 into $\zeta = z^\gamma$, $\zeta_1 = z_1^\gamma, \zeta_2 = \zeta/\zeta_1$ so that $\log_z z_1 = \log_\zeta \zeta_1$ holds.

Finally, the User picks $\tau \in \mathbb{Z}_q$ and computes $\eta = z^\tau$ (this will serve as an element of an additional Schnorr signature that proves knowledge of γ such that $\zeta = z^\gamma$).

Validation In this phase two Σ protocols are going to take place and combined according to the OR-proof. By (a, c, r) we are going to denote the transcript of Σ_y and by (a', c', r') we will denote the transcript of Σ_z. The Signer will compute Σ_z himself and Σ_y in interaction with the user.

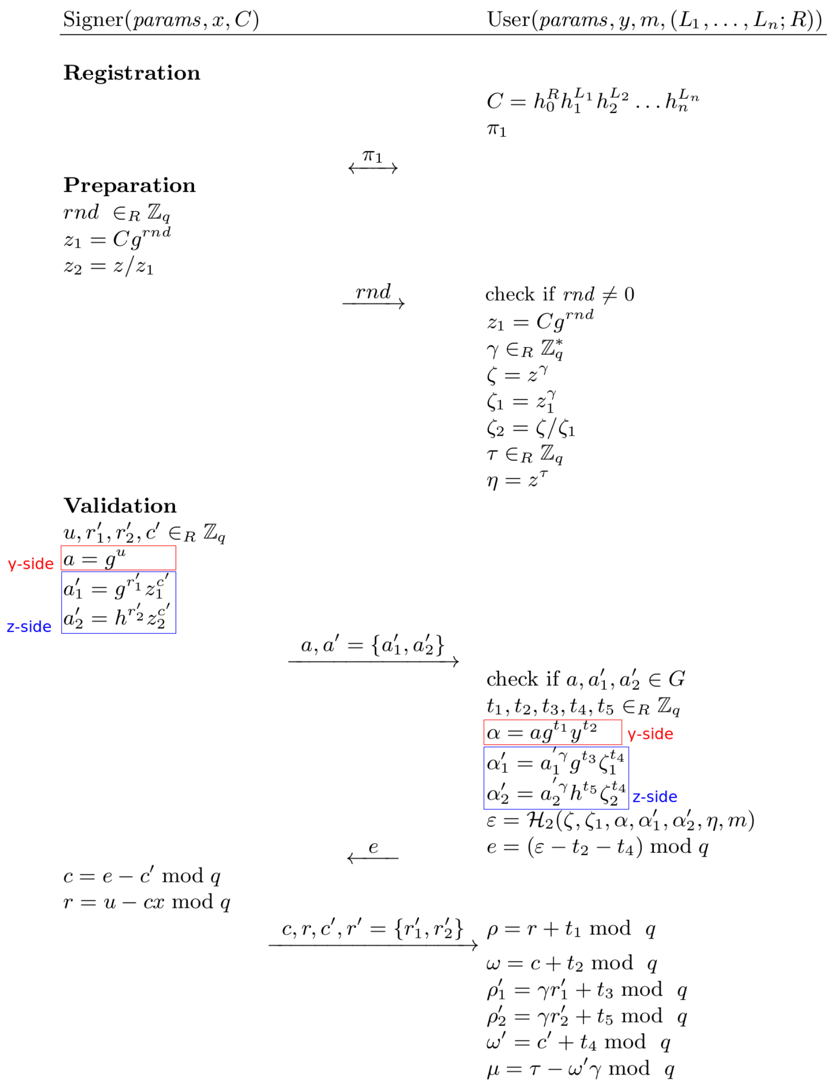
1. The Signer begins by creating a for Σ_y by picking a random $u \in \mathbb{Z}_q$ and computing $a = g^u$. Then, for Σ_z, following the OR-proof he picks random $c' \in \mathbb{Z}_q$ and $r' = \{r'_1, r'_2 \in \mathbb{Z}_q\}$. and sets $a'_1 = g^{r'_1} z_1^{c'}$ and $a'_2 = h^{r'_2} z_2^{c'}$. Finally he sends to the User $a, a' = \{a'_1, a'_2\}$.
2. The User checks whether $a, a'_1, a'_2 \in G$ and then picks blinding factors $t_1, t_2, t_3, t_4, t_5 \in_R \mathbb{Z}_q$ and blinds a into $\alpha = ag^{t_1}y^{t_2}$ and a'_1, a'_2 into $\alpha'_1 = a_1'^{\gamma} g^{t_3} \zeta_1^{t_4}$ and $\alpha'_2 = a_2'^{\gamma} h^{t_5} \zeta_2^{t_4}$. We denote $\alpha' = \{\alpha'_1, \alpha'_2\}$. Then, computes $\varepsilon = \mathcal{H}(\zeta, \zeta_1, \alpha, \alpha', \eta, m)$ where m is the message to be signed. The User sends to the Signer $e = (\varepsilon - t_2 - t_4) \bmod q$.
3. The Signer, according to the OR-proof technique, computes $c = e - c' \bmod q$ and $r = u - cx \bmod q$. Then, sends (c, r, c', r') to the User.

[2] Our scheme could be easily modified so that the issuer knows some of the user attributes. In that case the user would just send those attributes in clear to the issuer.

4. Finally, the User "unblinds" the received values and gets: $\rho = r + t_1 \bmod q$, $\omega = c + t_2 \bmod q$, $\rho'_1 = \gamma r'_1 + t_3 \bmod q$, $\rho'_2 = \gamma r'_2 + t_5 \bmod q$, $\omega' = c' + t_4 \bmod q$, $\mu = \tau - \omega'\gamma \bmod q$.

A signature is a 8-tuple $\sigma = (m, (\zeta, \zeta_1, \rho, \omega, \rho' = \{\rho'_1, \rho'_2\}, \omega', \mu))$ where ζ_1 encodes the attributes of the User ((ζ, ζ_1) corresponds to $\tilde{C}$, which is a blinded Pedersen commitment to $(L_1, \dots, L_n)$ with randomness (R, γ)).

In Figure 1 we describe all the phases of our ACL construction.



Signer$(params, x, C)$
User$(params, y, m, (L_1, \dots, L_n; R))$
Registration
$C = h_0^R h_1^{L_1} h_2^{L_2} \dots h_n^{L_n}$
π_1
$\xleftrightarrow{\pi_1}$
Preparation
$rnd \in_R \mathbb{Z}_q$
$z_1 = Cg^{rnd}$
$z_2 = z/z_1$
$\xrightarrow{rnd}$
check if $rnd \neq 0$
$z_1 = Cg^{rnd}$
$\gamma \in_R \mathbb{Z}_q^*$
$\zeta = z^\gamma$
$\zeta_1 = z_1^\gamma$
$\zeta_2 = \zeta/\zeta_1$
$\tau \in_R \mathbb{Z}_q$
$\eta = z^\tau$
Validation
$u, r'_1, r'_2, c' \in_R \mathbb{Z}_q$
y-side
$a = g^u$
$a'_1 = g^{r'_1} z_1^{c'}$
$a'_2 = h^{r'_2} z_2^{c'}$
z-side
$\xrightarrow{a, a' = \{a'_1, a'_2\}}$
check if $a, a'_1, a'_2 \in G$
$t_1, t_2, t_3, t_4, t_5 \in_R \mathbb{Z}_q$
$\alpha = ag^{t_1} y^{t_2}$
y-side
$\alpha'_1 = a_1'^{\gamma} g^{t_3} \zeta_1^{t_4}$
$\alpha'_2 = a_2'^{\gamma} h^{t_5} \zeta_2^{t_4}$
z-side
$\varepsilon = \mathcal{H}_2(\zeta, \zeta_1, \alpha, \alpha'_1, \alpha'_2, \eta, m)$
$e = (\varepsilon - t_2 - t_4) \bmod q$
$\xleftarrow{e}$
$c = e - c' \bmod q$
$r = u - cx \bmod q$
$\xrightarrow{c, r, c', r' = \{r'_1, r'_2\}}$
$\rho = r + t_1 \bmod q$
$\omega = c + t_2 \bmod q$
$\rho'_1 = \gamma r'_1 + t_3 \bmod q$
$\rho'_2 = \gamma r'_2 + t_5 \bmod q$
$\omega' = c' + t_4 \bmod q$
$\mu = \tau - \omega'\gamma \bmod q$

Fig. 1. Proposed ACL Construction

Verification A signature tuple (m, ζ_1, σ) (where ζ_1 corresponds to $\tilde{C}$) verifies if $\zeta \neq 1$ and

$$\omega + \omega' = \mathcal{H}(\zeta, \zeta_1, g^\rho y^\omega, g^{\rho'_1} \zeta_1^{\omega'}, h^{\rho'_2} \zeta_2^{\omega'}, z^\mu \zeta^{\omega'}, m) \bmod q.$$

5.3 Discussion

As briefly mentioned in the description of our construction there are certain "tricks" we can do to improve its efficiency. First of all notice that the *Registration* phase only needs to happen once for each User and a corresponding set of attributes. The Signer stores the attribute commitment C together with some identification information asked from the User (e.g. passport, ID).Then, it is sufficient if every time in the beginning of the *Signature Issuing* protocol the User identifies himself to the Signer in order to prove ownership of his account. Moreover, notice that *Preparation* and *Validation* phases can be combined and executed simultaneously. As a result, the whole signing protocol consists of three rounds.

Let's now discuss the differences between our ACL construction and the blind signature protocol described by Abe [1]. The basic advantage of ACL is that allows for the encoding of users' attributes in the signatures they receive from the Signer. In order for this to happen we need the *Registration* phase during which the User commits to his set of attributes. Those attributes are then encoded in the signature in ζ_1. The crucial difference from Abe's approach is that z_1 can no longer be computed as the result of some hash function given a random input *rnd*. In ACL, z_1 needs to include the attribute commitment C so $z_1 = Cg^{rnd}$ (we need the g^{rnd} factor so that two different signature issuings with the same user cannot be linked to each other). This is the step where the blinded signature inherits some structure from the values the signer can see in the signing step, and allows the encoding of attributes. The fact that z_1 is not the result of a hash function $\mathcal{H}'$ makes our security analysis different from Abe's. We can no longer define $\mathcal{H}'$ so that it will return the output we need, instead we will have to make use of ZK extractors. Unfortunately, we cannot take advantage of Abe's analysis because it only applied to special types of adversaries [33]. Of independent interest, our analysis indicates that, for sequential composition, Abe's blind signature may still be provably secure under DDH in the RO model. The formal analysis follows in the next section.

6 Proof of Security

The correctness of the scheme is straightforward, see Appendix B.

Theorem 3 (Blindness). *The proposed scheme satisfies blindness under the Decisional Diffie-Hellman assumption in the random oracle model.*

Proof. We wish to show that, when interacting with a challenger as described in the definition of blindness, the adversary cannot guess which signature is which (i.e. the adversary should not be able to tell which signature corresponds to each instance and each attribute set). We distinguish between two types of signatures that a challenger might output: a signature $\sigma = (m, (\zeta, \zeta_1, \rho, \omega, \rho_1', \rho_2', \omega', \mu))$ is *correct* for a particular interaction with the signer if there exists a value γ such that $\zeta = z^\gamma$ and $\zeta_1 = z_1^\gamma$ where z_1 is the value corresponding to this interaction with the signer, and the signature verifies. A signature is *fake* if no such γ exists. Note that it is easy for a challenger to generate both kinds of signatures: a correct signature can simply be output by correctly following the user's side of the protocol. With control of the random oracle, a fake signature is computed as follows: first, the challenger picks ζ and ζ_1 at random from G, and let $\zeta_2 = \zeta/\zeta_1$. Next, it picks random $\rho, \omega, \rho_1', \rho_2', \omega', \mu$ from $\mathbb{Z}_q$ and sets the random oracle such that $\omega + \omega' = \mathcal{H}(\zeta, \zeta_1, g^\rho y^\omega, g^{\rho_1'} \zeta_1^{\omega'}, h^{\rho_2'} \zeta_2^{\omega'}, z^\mu \zeta^{\omega'}, m)$, which ensures that the fake signature verifies.

We will prove the theorem by a hybrid argument. Consider the following three games in which the adversarial signer engages in two signature issuing instances with a challenger who outputs two signatures:

1. `Real`: The challenger outputs two correct signatures, σ_1, σ_2, in random order by honestly following the protocol. A correct signature is one that is honestly generated by interacting with the signer. Thus, in a correct signature the user blinds z, z_1 by the same γ so that $\log_z(z_1) = \log_\zeta(\zeta_1)$.
2. `Hybrid`: The challenger's output consists of a randomly picked correct signature and a fake signature, in random order. In a fake signature the challenger doesn't use the same random γ to blind z and z_1 so that $\log_z(z_1) \neq \log_\zeta(\zeta_1)$ except for negligible probability. Above, we

explained how a fake signature can be generated, assuming that the challenger controls the random oracle.

3. `Fake`: The challenger outputs two fake signatures.

What we need to prove is that `Real`$\approx$`Hybrid` and then that `Hybrid`$\approx$`Fake` (shown in Appendix C). Then, it holds that `Real` $\approx$ `Fake`. This proves the blindness of the ACL scheme since the view of the adversary is indistinguishable no matter if he receives two correct or two fake signatures.

Theorem 4 (One-More Unforgeability). *The signature issuing protocol is $(\ell, \ell{+}1)-$unforgeable for polynomially bounded ℓ if the discrete logarithm problem is intractable and $\mathcal{H}$ is a random oracle.*

Outline of the proof We first discuss the witness indistinguishability of the protocol (Lemma 1), which allows us to to simulate the Signer with either y-side or z-side witness(es) to extract the witness of the other side. Then, in Lemma 2 we prove that in order for the User to get a valid signature, he has to blind (z, z_1) into (ζ, ζ_1) only in such a way that $\log_z \zeta = \log_{z_1} \zeta_1$. In Lemma 3 we prove that is infeasible to create a valid signature without engaging in the issuing protocol with the legitimate Signer. From Lemmas 2 and 3 we see that if a User engages in the signature issuing protocol ℓ times and outputs $\ell + 1$ signatures, then, there exist at least two valid signatures linked to a particular run of the issuing protocol. Finally, it needs to be proven that a forger who manages to produce two signatures from a single protocol run can be used to solve the discrete logarithm problem.

We will denote the i-th execution of the issuing protocol by run_i. Recall that a transcript of run_i of the issuing protocol contains values $(z_{1,i}, z_{2,i})$; by the z-side witness of run_i we denote $(w_{1,i}, w_{2,i})$ such that $z_{1,i} = g^{w_{1,i}}$ and $z_{2,i} = h^{w_{2,i}}$.

Consider an alternative signing algorithm that, instead of using the y-side witness in the issuing protocol, uses the z-side witness. Let $params = (q, G, h, z, \{h_j\})$ be the public parameters, and y be the public key of the Signer. Suppose that the registration and the preparation phases of the signing protocol for run_i resulted in the signer and the user setting up the values $z_{1,i} = g^{w_{1,i}}$, $z_{2,i} = h^{w_{2,i}}$. The alternative signing algorithm takes as input the public parameters $params$, the public key y, the preparation phase output (z_1, z_2) and the values $(w_{1,i}, w_{2,i})$ instead of the secret key x, and works as follows:

1. Generate $c_i, r_i \in_U \mathbb{Z}_q$ and set $a_i := g^{r_i} y^{c_i}$.
2. Compute $a'_{1,i} := g^{u_{1,i}}$ and $a'_{2,i} := h^{u_{2,i}}$ with $u_{1,i}, u_{2,i} \in_U \mathbb{Z}_q$.
3. Sends $a_i, a'_{1,i}, a'_{2,i}$ to the user.
4. Given e_i from user, compute $c'_i := e_i - c_i \bmod q$, $r'_{1,i} := u_{1,i} - c'_i w_{1,i} \bmod q$ and $r'_{2,i} := u_{2,i} - c'_i w_{2,i} \bmod q$.
5. Send $r_i, c_i, r'_{1,i}, r'_{2,i}, c'_i$ to the user.

Definition 8. *The signing protocol described above is called* the z-side signer*.*

Lemma 1. *The signer's output is perfectly indistinguishable from the output of a z-side signer.*

Proof. Lemma 1 follows from the result of Cramer [17].

Note that, in order to run the z-side signer, it is necessary to somehow get the corresponding witness. We will see later that, with black-box access to an adversarial user in the registration phase, a reduction can use standard Σ-protocol extraction techniques in order to learn the representations of C; if the reduction is, in addition, given the discrete logarithms of all the public parameters to the base g, it will be able to compute the z-side witness, and so, inside a reduction, the z-side signer can be invoked.

Lemma 2. *(Restrictive Blinding) Let $\mathcal{A}$ be a User that engages in the signature issuing protocol ℓ times, and outputs a valid signature $\sigma = (m, (\zeta, \zeta_1, \rho, \omega, \rho'_1, \rho'_2, \omega', \mu))$. Let run_i denote the i-th execution of the protocol and $z_{1,i}$ denote z_1 used by the signer $\mathcal{S}$ in run_i. For polynomially bounded ℓ and for all polynomial-time $\mathcal{A}$, the probability that $\log_z \zeta \neq \log_{z_{1,i}} \zeta_1$ holds for all i, is negligible if the discrete logarithm problem is intractable and $\mathcal{H}$ is a random oracle.*

Proof. Let $\mathcal{A}$ have at most q_h accesses to $\mathcal{H}$ and ask at most ℓ signatures to $\mathcal{B}$. Let $\sigma = (m, (\zeta, \zeta_1, \rho, \omega, \rho'_1, \rho'_2, \omega', \mu))$ be a signature that $\mathcal{A}$ outputs and satisfies $\log_z \zeta \neq \log_{z_{1,i}} \zeta_1$ for all i with probability ϵ_0 which is not negligible in n (q_h and ℓ are bounded by a polynomial in the security parameter n). We randomly fix an index $I \in \{1, \ldots, \ell\}$ and regard $\mathcal{A}$ as successful only if the RO query for the resulting signature was during the I-th run. (If the resulting signature does not correspond to any query, then $\mathcal{A}$ is successful only with negligible probability due to the randomness of $\mathcal{H}$.) Let $(\mathbf{p}; \mathbf{q}; \mathbf{g}; \mathbf{Y})$ be an instance of the DL problem.

Reduction Algorithm: $\mathcal{B}$ first sets $(p, q, g) := (\mathbf{p}, \mathbf{q}, \mathbf{g})$. Then, it flips a coin $\chi \in_U \{0, 1\}$ to either select $y := \mathbf{Y}$ (case $\chi = 0$), or $h := \mathbf{Y}$ (case $\chi = 1$) and also guesses a random index $I \in \{1, \ldots, \ell\}$.

Case $y = \mathbf{Y}$: (extracting y-side witness)

1. *Key Generation*: $\mathcal{B}$ selects $k, k', \{k_j\} \in_U \mathbb{Z}_q$, for $0 \leq j \leq n$, and sets $h := g^k$ and $h_j := g^{k_j}$, $z = g^{k'}$. As a result of setting the parameters this way, $\mathcal{B}$ will always be able to compute z-side witnesses, as long as $\mathcal{B}$ can successfully extract the user's attributes and randomness in the registration phase.
2. *Registration:* $\mathcal{B}$ extracts values $(L_1, \ldots, L_n; R)$ using a knowledge extractor for proof π_1 (since we only worry about sequential composition the extractor may rewind) and defines $K = k_0 R + \sum_{ij1}^n k_j L_j$ (so that $C = g^K$).
3. *Preparation:* $\mathcal{B}$ selects $w_{1,i} \in \mathbb{Z}_q$ and sets $rnd = w_{1,i} - K$. Then, computes $w_{2,i} = (k' - w_{1,i})/k \bmod q$ (so we have $z_{1,i} = g^{w_{1,i}}, z_{2,i} = h^{w_{2,i}}$). $\mathcal{B}$ sends rnd to $\mathcal{A}$.
4. *Validation* $\mathcal{B}$ runs $\mathcal{A}$ using the z-side signer described above. At the end of each run_i the z-side signer sends: $r_i, c_i, r'_{1,i}, r'_{2,i}, c'_i$ to $\mathcal{A}$. $\mathcal{B}$ simulates $\mathcal{H}$ by returning $\varepsilon \in_U \mathbb{Z}_q$ to the random oracle queries issued by $\mathcal{A}$. $\mathcal{A}$ outputs a signature $\sigma = (\zeta, \zeta_1, \rho, \omega, \rho'_1, \rho'_2, \omega', \mu)$, that corresponds to some ε.
5. If the RO query for the resulting signature σ happened in run_I then move to the next step, otherwise the reduction fails.
6. *Rewinding*: Reset and restart $\mathcal{A}$ with the same setting. For run_I and after, $\mathcal{B}$ simulates $\mathcal{H}$ with $\tilde{\varepsilon} \in_U \mathbb{Z}_q$. $\mathcal{A}$ outputs a signature, say $\tilde{\sigma} = (\zeta, \zeta_1, \tilde{\rho}, \tilde{\omega}, \tilde{\rho'_1}, \tilde{\rho'_2}, \tilde{\omega'}, \tilde{\mu})$.
   If $\omega \neq \tilde{\omega}$, $\mathcal{B}$ outputs $x := (\rho - \tilde{\rho})/(\tilde{\omega} - \omega) \bmod q$. The simulation fails, otherwise.

Case $h = \mathbf{Y}$: (extracting z-side witness)

1. *Key generation*: $\mathcal{B}$ selects $x, \{k_j\} \in_U \mathbb{Z}_q$ and sets $y := g^x$ and $h_j := g^{k_j}$. It also selects $w_1, w_2 \in_U \mathbb{Z}_q$ and sets $z := g^{w_1} h^{w_2}$.
2. *Signature Issuing*: $\mathcal{B}$ runs $\mathcal{A}$ simulating the signer as follows.
   (a) For $i \neq I$, $\mathcal{B}$ follows the protocol with y-side witness, x.
   (b) For $i = I$, $\mathcal{B}$ engages in the issuing protocol using the z-side witness (w_1, w_2) as follows.
      i. *Registration* Extract values $(L_1, \ldots, L_n; R)$ using a ZK extractor and define $K = k_0 R + \sum_{i=1}^n k_i L_i$ (so that $C = g^K$).
      ii. *Preparation* Set $rnd = w_1 - K$ (so we have $z_{1,J} = g^{w_1}, z_{2,J} = h^{w_2}$). $\mathcal{B}$ sends rnd to $\mathcal{A}$.
      iii. *Validation* $\mathcal{B}$ runs $\mathcal{A}$ using the z-side signer and at the end of run_I sends $(r_I, c_I, r'_{1I}, r'_{2I}, c'_I)$ to $\mathcal{A}$.

   $\mathcal{B}$ simulates $\mathcal{H}$ by returning $\varepsilon \in_U \mathbb{Z}_q$ to the RO queries made by $\mathcal{A}$.
3. $\mathcal{A}$ outputs a signature, say $\sigma = (\zeta, \zeta_1, \rho, \omega, \rho'_1, \rho'_2, \omega', \mu)$, that corresponds to some ε. If the RO query for the forged signature didn't happen in run_I then fail, else move to the next step.
4. *Rewinding*: Rewind and restart $\mathcal{A}^*$ with the same setting. For run_I and forward, $\mathcal{B}$ simulates $\mathcal{H}$ by returning $\tilde{\varepsilon} \in_U \mathbb{Z}_q$.
5. $\mathcal{A}$ outputs a signature, say $\tilde{\sigma} = (\zeta, \zeta_1, \tilde{\rho}, \tilde{\omega}, \tilde{\rho'_1}, \tilde{\rho'_2}, \tilde{\omega'}, \tilde{\mu})$, that corresponds to some $\tilde{\varepsilon}$. If $\omega' \neq \tilde{\omega'}$, $\mathcal{B}$ computes $w'_1 = (\rho'_1 - \tilde{\rho'}_1)/(\mu - \tilde{\mu}) \bmod q$, $w'_2 = (\rho'_2 - \tilde{\rho'_2})/(\mu - \tilde{\mu}) \bmod q$, and outputs $w = (w_1 - w'_1)/(w'_2 - w_2) \bmod q$ (where $w = \log_g h$). Simulation fails otherwise.

Evaluation of success probability The reduction is successful when it gets $\omega \neq \tilde{\omega}$ for $\chi = 0$ or when it gets $\omega' \neq \tilde{\omega}'$ for $\chi = 1$. What we need to argue about is that, independently of the choice of

χ, the above will happen with non negligible probability. Note that we construct the reduction by giving different responses after the Ith run (for a randomly chosen $I \in \{1, \ldots, \ell\}$) when rewinding and we let the reduction fail if the forgery didn't happen in run_I.

Let us now fix the adversary's view and his RO tape. Once everything is fixed, we notice that $\mathcal{A}$'s forgery, is uniquely determined by c' and ε. We now consider two cases: (1) $\omega' = \omega$ with non negligible probability even if ε and c' change (after rewinding). In this case $\omega' = \tilde{\omega'}$ and thus $\omega \neq \tilde{\omega}$: $\omega + \omega' = \varepsilon \neq \tilde{\varepsilon} = \tilde{\omega} + \tilde{\omega'}$, so the y-side witness can be extracted. (2) $\omega' \neq \omega$ with non negligible probability if ε and c' change. So, when we run the adversary again with access to z-side witness and with different $\tilde{\varepsilon}$ we get two signatures from him where $\omega = \tilde{\omega}$ and thus $\omega' \neq \tilde{\omega'}$ and thus we can extract the z-side witness. Since we fail if the forgery didn't happen in run_I we do not need to consider what happens when we run $\mathcal{A}$ with y-side witness for $\chi = 1$ and $i \neq I$.

What is left to show is that for $\chi = 1$, the z-side witness, (w'_1, w'_2), extracted is not the one that B already knows Remember that the reduction in the beginning, selects $w_1, w_2 \in_U \mathbb{Z}_q$ and sets $z := g^{w_1} h^{w_2}$) and in run_I sets rnd so that $z_{1,I} = g^{w_1}, z_{2,I} = h^{w_2}$. The signature that $\mathcal{A}$ produces contains proofs that $\zeta = z^{\gamma}, \zeta_1 = g^{w'_1}$ and $\zeta_2 = h^{w'_2}$, so $w'_1 = \gamma w_1$ and $w'_2 = \gamma w_2$ and we can extract those w'_1, w'_2 by rewinding. For the forgery that $\mathcal{A}$ outputs it holds $\log_z \zeta \neq \log_{z_{1,I}} \zeta_1$ (by definition of Lemma 2) which guarantees that we will obtain two different representations of z.

Lemma 3. *Any poly-time adversary $\mathcal{A}$ outputs a valid signature without interacting with the Signer only with negligible probability if the discrete logarithm problem is intractable and $\mathcal{H}$ is a random oracle.*

The proof is straightforward and given in Appendix D.

Proof. (Theorem 4). Suppose that there exists adversary $\mathcal{A}$ that outputs $\ell + 1$ valid signatures with probability ϵ not negligible after interacting with the Signer at most ℓ times. The case $\ell = 0$ has been ruled out by Lemma 3. We consider $\ell \geq 1$.

By Lemmas 2 and 3, among the $\ell+1$ signatures, there exist at least two signature-message pairs which contain (ζ, ζ_1) and $(\bar{\zeta}, \bar{\zeta_1})$ such that $\log_{\zeta} \zeta_1 = \log_{\bar{\zeta}} \bar{\zeta_1} = \log_z z_{1I}$ holds for z_{1I} used in run_I for some I in $\{1, \ldots, \ell\}$. Now, there exist two queries to $\mathcal{H}$ that correspond to those signatures. In a similar way as in the proof of Lemma 2, we guess the indices of these queries and regard $\mathcal{A}$ as being successful only when the guess is correct. We consider an equivalent adversary $\mathcal{A}^*$ that asks $\mathcal{H}$ only twice and succeeds with probability $\epsilon^* = \epsilon / \binom{q_h}{2}$ in producing two signatures in the expected relation.

We construct a reduction $\mathcal{B}$ that, given $(\mathbf{p}, \mathbf{q}, \mathbf{g}, \mathbf{Y})$, solves $\log_g \mathbf{Y}$ in $\mathbb{Z}_q$ by using $\mathcal{A}^*$.

Reduction Algorithm: $\mathcal{B}$ sets $(p, q, g) := (\mathbf{p}, \mathbf{q}, \mathbf{g})$. It then flips a coin, $\chi \in_U \{0, 1\}$, to select either $y := \mathbf{Y}$ (case $\chi = 0$), or $y := g^x$ with randomly chosen x (case $\chi = 1$).

1. *Setup* If $\chi = 0$, set up and use the z-side signer. Set h, $h_0, \ldots, h_n$ such that their discrete logarithm base g is known. Else, set up the y-side signer by letting $y = g^x$ for a randomly chosen x; set up h, $h_0, \ldots, h_n$ such that their discrete logarithm base Y is known.
2. *Registration* If $\chi = 0$, extract $(R, L_0, \ldots, L_n)$. Else follow the protocol.
3. *Preparation* Follow the protocol.
4. *Validation* If $\chi = 0$, use the z-side signer, else follow the protocol.
5. *Responding to the RO queries* $\mathcal{B}$ simulates $\mathcal{H}$ by returning random values, say ε_1 and ε_2, to the two RO queries.
6. *Rewinding* When $\mathcal{A}^*$ outputs two signatures $(\sigma^{(1)} = (\zeta^{(1)}, \zeta_1^{(1)}, \rho^{(1)}, \omega^{(1)}, {\rho'}_1^{(1)}, {\rho'}_2^{(1)}, {\omega'}^{(1)}, \mu^{(1)}))$ and $(\sigma^{(2)} = (\zeta^{(2)}, \zeta_1^{(2)}, \rho^{(2)}, \omega^{(2)}, {\rho'}_1^{(2)}, {\rho'}_2^{(2)}, {\omega'}^{(2)}, \mu^{(2)}))$ corresponding to ε_1 and ε_2, $\mathcal{B}$ resets the adversary and this time responds to the two RO queries with $\tilde{\varepsilon}_1$ and $\tilde{\varepsilon}_2$.
7. *Interacting with $\mathcal{A}^*$ after rewinding* is the same as before, according to the value of χ.
8. *Extracting* When $\mathcal{A}^*$ again outputs two signatures $(\tilde{\sigma}^{(1)} = (\zeta^{(1)}, \zeta_1^{(1)}, \tilde{\rho}^{(1)}, \tilde{\omega}^{(1)}, \tilde{\rho'}_1^{(1)}, \tilde{\rho'}_2^{(1)}, \tilde{\omega'}^{(1)}, \tilde{\mu}^{(1)})$ and $(\tilde{\sigma}^{(2)} = (\zeta^{(2)}, \zeta_1^{(2)}, \tilde{\rho}^{(2)}, \tilde{\omega}^{(2)}, \tilde{\rho'}_1^{(2)}, \tilde{\rho'}_2^{(2)}, \tilde{\omega'}^{(2)}, \tilde{\mu}^{(2)})$ corresponding to $\tilde{\varepsilon}_1$ and $\tilde{\varepsilon}_2$, do: if $\chi = 0$ and $\omega^{(1)} \neq \tilde{\omega}^{(1)}$ or $\omega^{(2)} \neq \tilde{\omega}^{(2)}$, then extract the y-side witness as described in the proof of

Lemma 2; else if $\chi = 1$ and $\omega'^{(1)} \neq \tilde{\omega'}^{(1)}$ or $\omega'^{(2)} \neq \tilde{\omega'}^{(2)}$, then extract the z-side witness as described in Lemma 2.

Evaluation of Success probability Here we need to show that $\mathcal{B}$ will extract the witness that it doesn't already know with non-negligible probability (otherwise, χ is revealed). Consider all the q^2 possible $(\varepsilon_1, \varepsilon_2)$ pairs that can be given as responses to $\mathcal{A}^*$'s two RO queries. Let *succ* be the set of pairs $(\varepsilon_1, \varepsilon_2)$ for which $\mathcal{A}^*$ succeeds: $succ = \{(\varepsilon_1, \varepsilon_2) | \ \mathcal{A}$ breaks the seq. one-more unforgeability$\}$. What we want to show is that for a random pair $(\varepsilon_1, \varepsilon_2) \in succ$ there exists $(\tilde{\varepsilon_1}, \tilde{\varepsilon_2}) \in succ$ such that $\mathcal{A}^*$ produces the same e with non-negligible probability in the run where the first RO query happened. Notice that the transcript (i.e. the messages exchanged between the Signer and $\mathcal{A}^*$) of these two runs has to be the same. We call such a pair $(\tilde{\varepsilon_1}, \tilde{\varepsilon_2})$ *good* for $(\varepsilon_1, \varepsilon_2)$.

Consider $\mathcal{A}^*$ is run twice: once with $(\varepsilon_1, \varepsilon_2)$ and once with $(\tilde{\varepsilon_1}, \tilde{\varepsilon_2})$, both times producing two message-signature pairs corresponding to z_{1I}. Taken together, $(\varepsilon_1, \varepsilon_2)$, $(\tilde{\varepsilon_1}, \tilde{\varepsilon_2})$, the transcript of the runs and the four message-signature pairs allow the reduction to compute either a y side or a z side witness. However, we need to argue that, with non-negligible probability, this will be a witness the reduction doesn't already know. Since the transcripts are identical, the joint distribution of the two views ($\mathcal{A}^*$'s view in the first run and its view, after the rewinding, in the second run) is independent of χ.

First, note that there are only q possible e's but q^2 possible $(\tilde{\varepsilon_1}, \tilde{\varepsilon_2})$, and so there can only be a negligible fraction of $(\varepsilon_1, \varepsilon_2)$ for which no *good* $(\tilde{\varepsilon_1}, \tilde{\varepsilon_2})$ exists. Suppose that $(\tilde{\varepsilon_1}, \tilde{\varepsilon_2})$ is *good* for $(\varepsilon_1, \varepsilon_2)$ but $(\varepsilon_1^*, \varepsilon_2^*)$ is not; moreover, the reduction that gives $(\varepsilon_1, \varepsilon_2)$ before rewinding, and $(\varepsilon_1^*, \varepsilon_2^*)$ after rewinding, can only compute the witness it already knows (i.e. if $\chi = 0$, it can compute the z-side witness, and for $\chi = 1$ the y-side). Then for the same choice of χ, the reduction would be computing the witness it does *not* already know if, instead, it used $(\tilde{\varepsilon_1}, \tilde{\varepsilon_2})$ and $(\varepsilon_1^*, \varepsilon_2^*)$. For example, if $\chi = 0$ we would have $\omega \neq \tilde{\omega}$ and $\omega = \tilde{\omega^*}$ respectively, which implies that $\tilde{\omega} \neq \tilde{\omega^*}$ and thus by choosing $(\varepsilon_1, \varepsilon_2)$ and $(\varepsilon_1^*, \varepsilon_2^*)$ the y side witness could be extracted. Since it is just as likely to pick $(\varepsilon_1, \varepsilon_2)$ or $(\tilde{\varepsilon_1}, \tilde{\varepsilon_2})$ for the values before rewinding, it is just as likely to succeed as to fail, independently of χ (this analysis was first given by Abe and Okamoto [2]).

Finally, the only thing left to prove is that an adversary cannot open the sets of commitments $\{C^{(i)}\}$ and $\{\zeta^{(i)}, \zeta_1^{(i)}\}$ to two different multisets. Note that $\zeta^{(i)}$ uniquely determines $\gamma^{(i)}$, and by Lemma 2, it follows that for every i, there exists some $z_1^{(i')}$ such that $\zeta_1^{(i)} = (z_1^{(i')})^{\gamma^{(i)}}$. Thus, an alternative opening of the commitment $\{\zeta^{(i)}, \zeta_1^{(i)}\}$ gives rise to an alternative representation of $z_1^{(i')}$ in bases $h, h_1, \ldots, h_n, g$. In turn, by standard techniques, this breaks the discrete logarithm assumption. (The reduction will just honestly emulate the signer, and embed the DL instance into the parameters $h, h_1, \ldots, h_n, g$.)

References

1. Masayuki Abe. A secure three-move blind signature scheme for polynomially many signatures. In *EUROCRYPT*, pages 136–151, 2001.
2. Masayuki Abe and Tatsuaki Okamoto. Provably secure partially blind signatures. In *CRYPTO*, pages 271–286. Springer-Verlag, 2000.
3. Foteini Baldimtsi and Anna Lysyanskaya. On the security of one-witness blind signature schemes. In *Cryptology ePrint Archive, Report 2012/197*, 2012.
4. Mira Belenkiy, Melissa Chase, Markulf Kohlweiss, and Anna Lysyanskaya. Compact e-cash and simulatable vrfs revisited. In *Pairing '09*, pages 114–131, 2009.
5. Patrik Bichsel, Jan Camenisch, Thomas Groß, and Victor Shoup. Anonymous credentials on a standard java card. CCS '09, pages 600–610. ACM, 2009.
6. Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. PKC '03, pages 31–46, London, UK, UK, 2003. Springer-Verlag.
7. Stefan Brands. Untraceable off-line cash in wallets with observers. In *CRYPTO*, pages 302–318. Springer-Verlag, 1993.
8. Stefan A. Brands. Rethinking public key infrastructures and digital certificates: Building in privacy. MIT Press, Cambridge-London, August 2000.

9. Jan Camenisch and Thomas Groß. Efficient attributes for anonymous credentials. In *Proceedings of the 15th ACM conference on Computer and communications security*, CCS '08, 2008.
10. Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact e-cash. In *EUROCRYPT '05, volume 3494 of LNCS*, pages 302–321. Springer-Verlag, 2005.
11. Jan Camenisch, Maciej Koprowski, and Bodgan Warinschi. Efficient blind signatures without random oracles. SCN'04, pages 134–148, Berlin, Heidelberg, 2005. Springer-Verlag.
12. Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. EUROCRYPT '01, pages 93–118, London, UK, 2001. Springer-Verlag.
13. Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *Crypto 2004, LNCS 3152*, pages 56 – 72, 2004.
14. Jan Camenisch and Markus Michels. Proving in zero-knowledge that a number is the product of two safe primes. EUROCRYPT'99. Springer-Verlag, 1999.
15. Jan Camenisch, Fischer-Habner Simone, and Kai (eds.) Rannenberg. Privacy and identity management for life. In *ISBN 978-3-642-20316-9*. Springer, 2012.
16. David Chaum. Blind signatures for untraceable payment. In *Crypto'82*, pages 199–203, 1982.
17. Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO '94*, pages 174–187. Springer-Verlag, 1994.
18. Howard A. Schmidt (National cybersecurity coordinator). National strategy for trusted identities in cyberspace. In *Cyberwar Resources Guide, Item 163, http://www.projectcyw-d.org/resources/items/show/163*, 2010.
19. Ivan Damgård. Commitment schemes and zero-knowledge protocols. In *Lectures on Data Security, Modern Cryptology in Theory and Practice, Summer School, Aarhus, Denmark, July 1998*. Springer-Verlag, 1999.
20. Ivan Damgård. On σ- protocols. In *Course Notes, http://www.daimi.au.dk/~ivan/Sigma.ps*, 2002.
21. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO '86*, pages 186–194. Springer-Verlag, 1986.
22. Sanjam Garg, Vanishree Rao, Amit Sahai, Dominique Schröder, and Dominique Unruh. Round optimal blind signatures. In *CRYPTO*, pages 630–648, 2011.
23. O. Goldreich. *Foundations of Cryptography, vol. 1: Basic Tools*. Cambridge University Press, 2001.
24. S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18:186–208, February 1989.
25. Shafi Goldwasser and Yael Tauman Kalai. On the (in)security of the fiat-shamir paradigm. In *FOCS*, pages 102–113, 2003.
26. Jorge Guajardo, Bart Mennink, and Berry Schoenmakers. Anonymous credential schemes with encrypted attributes. In *Cryptology and Network Security*, pages 314–333. Springer, 2010.
27. Louis C. Guillou and Jean-Jacques Quisquater. A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. In *EUROCRYPT '88*, pages 123–128, 1988.
28. Nils Gura, Arun Patel, Arvinderpal Wander, Hans Eberle, and Sheueling Chang Shantz. Comparing elliptic curve cryptography and rsa on 8-bit cpus. In *CHES*, pages 119–132, 2004.
29. Gesine Hinterwalder, Christian T. Zenger, Foteini Baldimtsi, Anna Lysyanskaya, Christof Paar, and Wayne Burleson. Efficient e-cash in practice: Nfc-based payments for public transportation systems. In *PETS*, 2013.
30. Ari Juels, Michael Luby, and Rafail Ostrovsky. Security of blind digital signatures (extended abstract). In *CRYPTO*, pages 150–164. Springer-Verlag, 1997.
31. Marc Langheinrich. Privacy by design principles of privacy-aware ubiquitous systems. In *Ubicomp 2001: Ubiquitous Computing*, volume 2201, pages 273–291. Springer Berlin, Heidelberg, 2001.
32. Alfred J. Menezes, Paul C. Van Oorschot, and Scott A. Vanstone. Handbook of applied cryptography, 1997.
33. Miyako Ohkubo and Masayuki Abe. Security of three-move blind signature schemes reconsidered. In *SCIS'03, Symposium on Cryptography and Information Security, Japan*, 2003.
34. Tatsuaki Okamoto. Efficient blind and partially blind signatures without random oracles. In *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pages 80–99. Springer, 2006.
35. Christian Paquin. U-prove cryptographic specification v1.1. In *Microsoft Technical Report, http://connect.microsoft.com/site1188*, February 2011.
36. European Parliament and Council of the European Union. Regulation (ec) 45/2001. In *Official Journal of the European Union*, 2001.
37. European Parliament and Council of the European Union. Directive 2009/136/ec. In *Official Journal of the European Union*, 2009.
38. Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. CRYPTO '91. Springer-Verlag.
39. David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. In *Journal Of Cryptology*, volume 13, pages 361–396, 2000.
40. David Pointcheval and Jacques Stern. Provably secure blind signature schemes. In *Asiacrypt '96, LNCS 1163*, pages 252–265. Springer-Verlag, Feb 2011.
41. Claus P. Schnorr. Efficient identification and signatures for smart cards. CRYPTO '89, pages 239–252. Springer-Verlag New York, Inc., 1989.
42. IBM Security Team. Specification of the identity mixer cryptographic library, version 2.3.0. In *IBM Research Report*, 2010.
43. Gummeson J. Ransford B. Zhang, H. and K. Fu. Moo: A batteryless computational rfid and sensing platform. In *Tech. Rep. UM-CS-2011-020, UMass Amherst Department of Computer Science*, 2011.

A Basic Assumptions

The following two standard cryptographic assumptions are useful in the paper.

Definition 9. *(Discrete Logarithm Assumption) Let k be the security parameter. Let G_k be a set of groups such that G_k is a group of order q (k-bit prime) and generator g. Then, for every polynomial time algorithm $\mathcal{A}$ it holds that:*

$$Pr[h \leftarrow G; x \leftarrow A(h) : x = \log_g h] \leq \nu(k)$$

where $\nu(k)$ is a negligible function.

Definition 10. *(Decisional Diffie-Hellman Assumption) Let DH be an instance generator that, on input the security parameter 1^k outputs (G, q, g, A, B, C) where G is a cyclic group of order q (k-bit prime) with generator g, and, for randomly chosen $a, b \in \mathbb{Z}_q$, $A = g^a$, $B = g^b$ and $C = g^{ab}$. Let Rand be an instance generator that, on input 1^k outputs (G, q, g, A, B, C) where G,q,g,A and B are as above, while $C = g^c$ for a random $c \in \mathbb{Z}_q$. We assume that DH and Rand are computationally indistinguishable.*

B Correctness

The correctness of the new scheme is easily verified:

$$\begin{aligned}
\omega + \omega' &= c + t_2 + c' + t_4 = e + t_2 + t_4 = \varepsilon \bmod q \\
g^{\rho} y^{\omega} &= g^{r+t_1} y^{c+t_2} = g^{r+cx} g^{t_1} y^{t_2} = \alpha \\
g^{\rho'_1} \zeta_1^{\omega'} &= g^{\gamma r'_1 + t_3} \zeta_1^{c'+t_4} = (a'_1 z_1^{-\omega'})^{\gamma} g^{t_3} \zeta_1^{c'+t_4} \\
&= a_1'^{\gamma} g^{t_3} \zeta_1^{t_4} = \alpha'_1 \\
g^{\rho'_2} \zeta_2^{\omega'} &= h^{\gamma r'_2 + t_5} \zeta_2^{c'+t_4} = (a'_2 z_2^{-\omega'})^{\gamma} h^{t_5} \zeta_2^{c'+t_4} \\
&= a_2'^{\gamma} h^{t_5} \zeta_2^{t_4} = \alpha'_2 \\
z^{\mu} \zeta^{\omega'} &= z^{\tau - \omega' \gamma} \zeta^{\omega'} = \zeta^{\gamma} = \eta.
\end{aligned}$$

C Blindness

In order to conclude the proof of Theorem 3 we will first show that `Real`$\approx$`Hybrid` and then that `Hybrid`$\approx$`Fake`.

Proof. (*contin.*) **Case 1:** `Real`$\approx$`Hybrid`

Suppose that there exists a poly-time adversary $\mathcal{A}$ who is successful in distinguishing between `Real` and `Hybrid` with probability $1/2 + \epsilon$, where the advantage ϵ is not negligible. We then show that there exists a poly-time algorithm $\mathcal{B}$ that solves the DDH problem with non-negligible advantage.

The reduction $\mathcal{B}$ gets as input an instance (g, A, B, D) where (g, A, B, D) is a DDH instance. $\mathcal{B}$ first fixes $z = A$ and then sets $h_0 = g^{e_0}$, and $h_1, \ldots, h_n$ to $A^{e_0}, \ldots, A^{e_n}$ for randomly chosen $e_i \in \mathbb{Z}_q$. $\mathcal{B}$ sends the parameters to $\mathcal{A}$. The adversary $\mathcal{A}$ creates his public key y and sends y to $\mathcal{B}$ along with messages m_0, m_1, commitments C_0, C_1 and openings $(\overrightarrow{L_0}, \overrightarrow{L_1})$ which are two attributes vectors of size n. $\mathcal{A}$ and $\mathcal{B}$ engage in two interleaving registration and signature issuing instances. During the registration phase we have: $C_{(j)} = A^{k_1^{(j)}}$, ($k_1^{(j)} = e_0 R + e_1 L_{j,1} + \cdots + e_n L_{j,n}$) honestly created, for $j \in \{0, 1\}$. Note that the value $k_1^{(j)}$ is known to the reduction $\mathcal{B}$.

Then, during the two signature issuing instances, $\mathcal{A}$ computes $z_1^{(j)} = A^{k_1^{(j)}} g^{rnd^{(j)}}$ where $rnd^{(j)}$ was sent to $\mathcal{B}$. $\mathcal{B}$ responds with random $e^{(j)}$ and then computes two signatures σ_1, σ_2 in the following way:

$\mathcal{B}$ flips two random coins: $coin, b \in \{0,1\}$. Then, lets σ_{2-coin} be the correct signature on $(m_0, \tilde{C}_b)$ (where $\tilde{C}$ is denoted as ζ_1 in our ACL scheme) for the corresponding instance, and σ_{coin+1} be a signature on $(m_1, \tilde{C}_{1-b})$ generated from the input to $\mathcal{B}$, as follows: set $\zeta^{(j)} = D$ and $\zeta_1^{(j)} = D^{k_1^{(j)}} B^{rnd^{(j)}}$; let $\zeta_2^{(j)} = \zeta^{(j)}/\zeta_1^{(j)}$, randomly choose $\rho, \omega, \rho_1', \rho_2', \omega', \mu$, and then define $\mathcal{H}$ so that the signature verifies.

After receiving the signatures, $\mathcal{A}$ outputs `Real` or `Hybrid`. If $\mathcal{A}$ outputs `Real` then $\mathcal{B}$ outputs "DH" or $\mathcal{B}$ outputs "random" otherwise.

Let us analyze the success of $\mathcal{B}$ in distinguishing Diffie-Hellman tuples from random tuples. If the input (g, A, B, D) is a DH tuple, then: $\zeta^{(j)} = D = g^{ab} = A^b = z^b$ and $\zeta_1^{(j)} = D^{k_1^{(j)}} B^{rnd^{(j)}} = g^{abk_1^{(j)}} g^{brnd^{(j)}} = (g^{ak_1^{(j)}} g^{rnd^{(j)}})^b = (A^{k_1^{(j)}} g^{rnd^{(j)}})^b = z_1^b$ so, the signature σ_{coin+1} is distributed identically to a correct signature, and this is precisely the `Real` game. Similarly, if the input is not a DH tuple then it is the `Hybrid` game. Therefore, $\mathcal{B}$ will be correct exactly when $\mathcal{A}$ is and will distinguish a DH tuple with probability $1/2 + \epsilon$, which contradicts to the DDH assumption.

Case 2: `Hybrid`$\approx$`Fake`
Similarly, we build a reduction $\mathcal{B}$ which given an adversary $\mathcal{A}$, who distinguishes between `Hybrid` and `Fake` with probability $1/2 + \epsilon$, solves the DDH problem.

Working in a similar way we did before, $\mathcal{B}$ will now create a fake signature and one using the input tuple. If $\mathcal{A}$ outputs `Hybrid` then $\mathcal{B}$ will output "DH" or "not DH" otherwise. If the input is a DH instance then the output will correspond to the `Hybrid` game and to the `Fake` otherwise. So, $\mathcal{B}$ will be successful whenever $\mathcal{A}$ is and will break the DDH assumption with non-negligible advantage ϵ which is a contradiction.

Finally, since `Real`$\approx$`Hybrid` and `Hybrid`$\approx$`Fake` it follows that `Real`$\approx$`Fake`.

D Proof of Lemma 3

Proof. (sketch) In order to prove this we are going to use the *oracle replay attack* suggested by Pointcheval and Stern to prove unforgeability of blind signatures [40]. Similar to the proof of Lemma 2 we randomly fix an index $I \in \{1, \ldots, \ell\}$ and regard $\mathcal{A}$ as successful only if the RO query for the resulting signature was during the I-th run. Accordingly, it is equivalent to assuming an adversary, we will call it $\mathcal{A}^*$, that asks $\mathcal{H}$ only once and succeeds with probability $\epsilon_1 \geq \epsilon_0/q_h$. Given this modified $\mathcal{A}^*$ we construct the reduction $\mathcal{B}$ that solves the discrete logarithm problem by simulating the interaction. Given $\mathbf{Y} \in G$, we construct a reduction $\mathcal{B}$ that finds $\log_g \mathbf{Y}$ in $\mathbb{Z}_q$. During *key generation*, $\mathcal{B}$ first selects $\omega, \xi, \{k_i\}$ randomly and sets $y = \mathbf{Y}, h = g^w, h_i = g^{k_i} z = \mathbf{Y} g^{\xi}$. ($\mathcal{B}$ does not need to simulate the signer, so it can put $\mathbf{Y}$ in both y and z.) Then, $\mathcal{B}$ invokes $\mathcal{A}^*$ twice with the same initial settings and different ε and $\tilde{\varepsilon}$ as answers of $\mathcal{H}$. The resulting signatures are $\sigma = (\zeta, \zeta_1, \rho, \omega, \rho_1', \rho_2', \omega', \mu)$ and $\tilde{\sigma} = (\zeta, \zeta_1, \tilde{\rho}, \tilde{\omega}, \tilde{\rho_1'}, \tilde{\rho_2'}, \tilde{\omega'}, \tilde{\mu})$. Since $\omega + \omega' = \varepsilon \neq \tilde{\varepsilon} = \tilde{\omega} + \tilde{\omega'}$, at least one of $\omega \neq \tilde{\omega}$ or $\omega' \neq \tilde{\omega'}$ happens.

In the case that $\omega \neq \tilde{\omega}$, $\mathcal{B}$ computes $\log_g \mathbf{Y} = (\rho - \tilde{\rho})/(\tilde{\omega} - \omega) \bmod q$. For the case $\omega' \neq \tilde{\omega'}$, $\mathcal{B}$ computes $\gamma = \log_z \zeta = (\mu - \tilde{\mu})/(\omega' - \tilde{\omega'}) \bmod q$, $w_1 = \log_g \zeta_1 = (\rho_1' - \tilde{\rho_1'})/(\omega' - \tilde{\omega'}) \bmod q$, $w_2 = \log_g \zeta_2 = (\rho_2' - \tilde{\rho_2'})/(\omega' - \tilde{\omega'}) \bmod q$, and $\log_g \mathbf{Y} = \log_g z - \xi = (w_1 + w_2/w)/\gamma - \xi \bmod q$.