Signature Schemes and Anonymous Credentials from Bilinear Maps

Jan Camenisch
IBM Research
Zurich Research Laboratory
CH–8803 Rüschlikon
jca@zurich.ibm.com

Anna Lysyanskaya
Computer Science Department
Brown University
Providence, RI 02912 USA
anna@cs.brown.edu

Abstract. We propose a new and efficient signature scheme that is provably secure in the plain model. The security of our scheme is based on a discrete-logarithm-based assumption put forth by Lysyanskaya, Rivest, Sahai, and Wolf (LRSW) who also showed that it holds for generic groups and is independent of the decisional Diffie-Hellman assumption. We prove security of our scheme under the LRSW assumption for groups with bilinear maps. We then show how our scheme can be used to construct efficient anonymous credential systems as well as group signature and identity escrow schemes. To this end, we provide efficient protocols that allow one to prove in zero-knowledge the knowledge of a signature on a committed (or encrypted) message and to obtain a signature on a committed message.

1 Introduction

Digital signatures schemes, invented by Diffie and Hellman [20], and formalized by Goldwasser, Micali and Rivest [26], not only provide the electronic equivalent of signing a paper document with a pen but also are an important building block for many cryptographic protocols such as anonymous voting schemes, e-cash, and anonymous credential schemes, to name just a few.

Signature schemes exists if and only if one-way functions exist [32,35]. However, the efficiency of these general constructions, and also the fact that these signature schemes require the signer's secret key to change between invocations of the signing algorithm, makes these solutions undesirable in practice.

Using an ideal random function (this is the so-called *random-oracle* model), several, much more efficient signature schemes were shown to be secure. Most notably, those are the RSA [34], the Fiat-Shamir [21], and the Schnorr [36] signature schemes. However, ideal random functions cannot be implemented in the plain model [13,25], and therefore in the plain model, these signature schemes are not provably secure.

Over the years, many researchers have come up with signature schemes that are efficient and at the same time provably secure in the plain model. The most efficient ones provably secure in the standard model are based on the strong RSA assumption [23,19,22,10]. However, no scheme based on an assumption related to the discrete logarithm assumption in the plain (as opposed to random-oracle) model comes close to the efficiency of these schemes.

In this paper, we propose a new signature scheme that is based on an assumption introduced by Lysyanskaya, Rivest, Sahai, and Wolf [30] and uses bilinear maps. This assumption was shown to hold for generic groups [30], and be independent of the decisional Diffie-Hellman assumption. Our signature scheme's efficiency is comparable to the schemes mentioned above that are based on the Strong RSA assumption.

We further extend our basic signature scheme such that it can be used as a building block for cryptographic protocols. To this end, we provide protocols to prove knowledge of a signature on a committed message and to obtain a signature on a committed message. These protocols yield a group signature scheme [17] or an anonymous credential system [14] (cf. [10]). That is, we obtain the first efficient and secure credential system and group signature/identity escrow schemes [28] that are based solely on discrete-logarithm-related assumptions. We should mention that an anonymous credential system proposed by Verheul [38] is also only based on discrete logarithm related assumptions; however, the scheme is not proven secure. Also note that the recent scheme by Ateniese and de Medeiros [2] requires the strong RSA assumption although no party is required to know an RSA secret key during the operation of the system.

Note that not only are our group signature and anonymous credential schemes interesting because they are based on a different assumption, but also because they are much more efficient than any of the existing schemes. All prior schemes [1,9,10,2] required proofs of knowledge of representations over groups modulo large moduli (for example, modulo an RSA modulus, whose recommended length is about 2K Bits).

Recently, independently from our work, Boneh and Boyen [4] put forth a signature scheme that is also provably secure under a discrete-logarithm-type assumption about groups with bilinear maps. In contrast to their work, our main goal is not just an efficient signature scheme, but a set of efficient protocols to prove knowledge of signatures and to issue signatures on committed (secret) messages. Our end goal is higher-level applications, i.e., group signature and anonymous credential schemes that can be constructed based solely on an assumption related to the discrete logarithm assumption.

In another recent independent work, Boneh, Boyen, and Shacham [5] construct a group signature scheme based on different discrete-logarithm-type assumptions about groups with bilinear pairings. Their scheme yields itself to the design of a signature scheme with efficient protocols as well. In §5 we describe their scheme and its connection to our work in more detail.

Outline of the paper. In §2 we give our notation and some number-theoretic preliminaries, including bilinear maps and the LRSW assumption. In §3, we give our signature scheme and prove it secure. In §4 we show how our signature yields itself to the design of an anonymous credential system: we give protocols for obtaining a signature on a committed value, and for proving knowledge of a signature on a committed value. In the end of that section, we show how to realize a group signature scheme based on our new signature. Finally, in Section 5, we show that the scheme of Boneh, Boeyn and Shacham can be extended so that a signature scheme with efficient protocols, similar to the one we describe in Sections 3 and 4 can be obtained based on their assumptions as well.

2 Preliminaries

We use notation introduced by Micali [31] (also called the GMR notation), and also notation introduced by Camenisch and Stadler [12]. Here we review it briefly; the complete description can be found in the full version [CL04] of this paper.

If A is an algorithm, and b be a Boolean function, then by $(y \leftarrow A(x) : b(y))$, we denote the event that $b(y) = 1$ after y was generated by running A on input x. By $A^O(\cdot)$, we denote a Turing machine that makes queries to an oracle O. By $Q = Q(A^O(x)) \leftarrow A^O(x)$ we denote the contents of the query tape once A terminates, with oracle O and input x.

A function $\nu(k)$ is *negligible* if for every positive polynomial $p(\cdot)$ and for sufficiently large k, $\nu(k) < \frac{1}{p(k)}$.

Camenisch and Stadler[12] introduced notation for various proofs of knowledge of discrete logarithms and proofs of the validity of statements about discrete logarithms. For instance,

$$PK\{(\alpha, \beta, \gamma) : y = g^\alpha h^\beta \ \wedge \ \tilde{y} = \tilde{g}^\alpha \tilde{h}^\gamma \ \wedge \ (u \le \alpha \le v)\}$$

denotes a "*zero-knowledge* Proof *of* Knowledge *of integers* α, β, *and* γ *such that* $y = g^\alpha h^\beta$ *and* $\tilde{y} = \tilde{g}^\alpha \tilde{h}^\gamma$ *holds, where* $u \le \alpha \le v$," where $y, g, h, \tilde{y}, \tilde{g}$, and $\tilde{h}$ are elements of some groups $G = \langle g \rangle = \langle h \rangle$ and $\tilde{G} = \langle \tilde{g} \rangle = \langle \tilde{h} \rangle$. The convention is that Greek letters denote quantities the knowledge of which is being proved, while all other parameters are known to the verifier. We will sometimes apply the Fiat-Shamir heuristic to turn such a proof into a signature on a message m, which we will denote as, e.g., $SPK\{(\alpha) : y = g^\alpha\}(m)$.

We also use the standard definition of a digital signature scheme [26].

2.1 Number-Theoretic Preliminaries

We now describe some number-theoretic preliminaries. Suppose that we have a setup algorithm *Setup* that, on input the security parameter 1^k, outputs the setup for $G = \langle g \rangle$ and $\mathsf{G} = \langle \mathsf{g} \rangle$, two groups of prime order $q = \Theta(2^k)$ that have a non-degenerate efficiently computable bilinear map e. More precisely: We assume that associated with each group element, there is a unique binary string that represents it. (For example, if $G = \mathbb{Z}_p^*$, then an element of G can be represented as an integer between 1 and $p - 1$.) Following prior work (for example, Boneh and Franklin [6]), e is a function, $e \ : \ G \times G \rightarrow \mathsf{G}$, such that

- (Bilinear) For all $P, Q \in G$, for all $a, b \in \mathbb{Z}$, $e(P^a, Q^b) = e(P, Q)^{ab}$.
- (Non-degenerate) There exists some $P, Q \in G$ such that $e(P, Q) \neq 1$, where 1 is the identity of G.
- (Efficient) There exists an efficient algorithm for computing e.

We write: $(q, G, \mathsf{G}, g, \mathsf{g}, e) \leftarrow \mathit{Setup}(1^k)$. It is easy to see, from the first two properties, and from the fact that G and G are both of the same prime order q, that whenever g is a generator of G, $\mathsf{g} = e(g, g)$ is a generator of G.

Such groups, based on the Weil and Tate pairings over elliptic curves (see Silverman [37]), have been extensively relied upon in cryptographic literature over the past few years (cf. [27,6,7,24] to name a few results).

Further, we make the following assumption about the groups G and G.

Assumption 21 (LRSW Assumption) *Suppose that $G = \langle g \rangle$ is a group chosen by the setup algorithm Setup. Let $X, Y \in G$, $X = g^x$, $Y = g^y$. Let $O_{X,Y}(\cdot)$ be an oracle that, on input a value $m \in \mathbb{Z}_q$, outputs a triple $A = (a, a^y, a^{x+mxy})$ for a randomly chosen a. Then for all probabilistic polynomial time adversaries $\mathcal{A}^?$, $\nu(k)$ defined as follows is a negligible function:*

$$\begin{aligned}\Pr[&(q, G, \mathsf{G}, g, \mathsf{g}, e) \leftarrow \mathit{Setup}(1^k); x \leftarrow \mathbb{Z}_q; y \leftarrow \mathbb{Z}_q; X = g^x; Y = g^y;\\ &(m, a, b, c) \leftarrow \mathcal{A}^{O_{X,Y}}(q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y) \;:\; m \notin Q \;\wedge\; m \in \mathbb{Z}_q \wedge\\ &\wedge\; m \neq 0 \wedge\; a \in G \wedge\; b = a^y \;\wedge\; c = a^{x+mxy}] = \nu(k)\ ,\end{aligned}$$

where Q is the set of queries that $\mathcal{A}$ made to $O_{X,Y}(\cdot)$.

This assumption was introduced by Lysyanskaya et al. [30], and considered for groups that are not known to admit an efficient bilinear map. It was also shown, in the same paper, that this assumption holds for generic groups. It is not hard to see that the proof carries over to generic groups G and G with a bilinear map between them.

3 Three Signature Schemes

First, we present a simple signature scheme (Scheme A) and prove it secure under the LRSW assumption. Then, we modify this scheme to get signature schemes that lend themselves more easily to the design of efficient protocols for issuing a signature on a committed value and proving knowledge of a signature on a committed value. The first generalization will allow to sign such that the signature produced is independent of the message (Scheme B), which we generalize further into a scheme that allows to sign blocks of messages (Scheme C).

Schemes A and B are, in fact, special cases of Scheme C. So we really propose just *one* new signature scheme, namely Scheme C. Schemes A and B are just steps that simplify our presentation by making it more modular.

3.1 Scheme A: A Simple Signature Scheme

The signature scheme consists of the following algorithms:

Key generation. The key generation algorithm runs the *Setup* algorithm in order to generate $(q, G, \mathsf{G}, g, \mathsf{g}, e)$. It then chooses $x \leftarrow \mathbb{Z}_q$ and $y \leftarrow \mathbb{Z}_q$, and sets $sk = (x, y)$, $pk = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y)$, where $X = g^x$ and $Y = g^y$.

Signature. On input message m, secret key $sk = (x, y)$, and public key $pk = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y)$, choose a random $a \in G$, and output the signature $\sigma = (a, a^y, a^{x+mxy})$.

Verification. On input $pk = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y)$, message m, and purported signature $\sigma = (a, b, c)$, check that the following verification equations hold.

$$e(a, Y) = e(g, b) \qquad \text{and} \qquad e(X, a) \cdot e(X, b)^m = e(g, c) \ . \tag{1}$$

Theorem 1. *Signature Scheme A described above is correct and secure under the LRSW assumption.*

Proof. We first show correctness. The first verification equation holds as $e(a, Y) = e(a, g)^y = e(g, a)^y = e(g, b)$ and the second one holds because $e(X, a) \cdot e(X, b)^m = e(g, a)^x \cdot e(g, a)^{mxy} = e(g, a)^{x+mxy} = e(g, c)$.

We now show security. Without loss of generality, let $\mathsf{g} = e(g, g)$.

Consider the adversary interacting with the signer and outputting a valid signature σ on some message m that he did not query for. It is clear that the signer acts the same way as the oracle $O_{X,Y}$ defined in the LRSW assumption. Therefore, in order to prove security, we must show that the forgery $\sigma = (a, b, c)$ that passes the verification equations, must be of the form (*) $b = a^y$ and (**) $c = a^{x+mxy}$.

Let $a = g^\alpha$, $b = g^\beta$, $c = g^\gamma$. So, we wish to show that $\beta/\alpha = y$, and that $\gamma/\alpha = x + mxy$.

From the first verification equation and the bilinearity of e, we get that

$$\mathsf{g}^{\alpha y} = e(g, g)^{\alpha y} = e(a, Y) = e(g, b) = e(g, g)^\beta = \mathsf{g}^\beta \ .$$

As g is a generator of G, we can take the logarithm base g on both sides, and obtain $\alpha y = \beta \bmod q$, which gives us (*) as desired.

From the second verification equation, using the above, and, again, the fact that g is a generator:

$$\begin{aligned} e(X, a) \cdot e(X, b)^m &= e(g, c) \\ e(g, g)^{x\alpha} e(g, g)^{mx\beta} &= e(g, g)^\gamma \\ x\alpha + mx\beta \ = \ \alpha(x + mxy) &= \gamma \ . \end{aligned}$$

3.2 Scheme B Where Signature Is Independent of the Message

For constructing anonymous credentials, we need a signature scheme where the signature itself is distributed in a way that is information-theoretically independent of the message m being signed. In essence, what is being signed should be an information-theoretically secure commitment (Pedersen commitment) of the message. Thus, we modify Scheme A and obtain Scheme B as follows:

Key generation. Run the *Setup* algorithm to generate $(q, G, \mathsf{G}, g, \mathsf{g}, e)$. Choose $x \leftarrow \mathbb{Z}_q$, $y \leftarrow \mathbb{Z}_q$, $z \leftarrow \mathbb{Z}_q$. Let $X = g^x$, $Y = g^y$ and $Z = g^z$. Set $sk = (x, y, z)$, $pk = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y, Z)$.

Signature. On input message (m, r), secret key $sk = (x, y, z)$, and public key $pk = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y, Z)$ do:
- Choose a random $a \leftarrow G$.
- Let $A = a^z$.
- Let $b = a^y$, $B = A^y$.
- Let $c = a^{x+xym} A^{xyr}$.

Output $\sigma = (a, A, b, B, c)$.

Verification. On input $pk = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y, Z)$, message (m, r), and purported signature $\sigma = (a, A, b, B, c)$, check the following:
1. A was formed correctly: $e(a, Z) = e(g, A)$.
2. b and B were formed correctly: $e(a, Y) = e(g, b)$ and $e(A, Y) = e(g, B)$.
3. c was formed correctly: $e(X, a) \cdot e(X, b)^m \cdot e(X, B)^r = e(g, c)$.

Note that the values $(g^m Z^r, a, A, b, B, c)$ are information-theoretically independent of m if r is chosen randomly. This will become crucial when using this signature scheme in the context of an anonymous credential system.

Theorem 2. *Signature Scheme B described above is correct and secure under the LRSW assumption.*

The full proof of this theorem is found in the full version [CL04] of this paper. Here we give a sketch. Correctness follows by inspection. To show security, we consider two types of forgery. Type 1 forgery is on some message (m, r) such that for all previously queried (m_i, r_i) we have $g^m Z^r \neq g^{m_i} Z^{r_i}$. Type 2 forgery is when this is not the case.

The existence of Type-1 forger contradicts the LRSW assumption by reduction from Signature Scheme A. On input a public key $pk = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y)$ for Scheme A, our reduction forms a public key $pk' = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y, Z)$ for Scheme B by choosing $z \leftarrow \mathbb{Z}_q$ and setting $Z = g^z$. It then runs the forger on input pk', and answers signature queries of the form (m_i, r_i) by transforming them into queries $m'_i = m_i + r_i z \bmod q$ for the signature oracle for Scheme A. It is easy to see that a Type 1 forgery on (m, r) constitutes a successful forgery for the message $m' = m + rz$ in Scheme A.

The existence of Type-2 forger contradicts the discrete logarithm assumption (and therefore the LRSW assumption). The reduction takes as input $(q, G, \mathsf{G}, g, \mathsf{g}, e, Z)$, and sets up the public key for the signature scheme by choosing X and Y. It then runs the forger, answers all the signature queries (since it generated X and Y itself) and obtains a Type-2 forgery, namely (m, r), (m_i, r_i) such that $g^m Z^r = g^{m_i} Z^{r_i}$ for some i. This immediately gives the discrete logarithm of Z to the base g.

3.3 Scheme C for Blocks of Messages

Scheme B allows us to generate a signature on m in such a way that the signature itself reveals no information about m. Namely, one can choose a random r and sign (m, r) using Scheme B. In general, however, there is no reason that we should limit ourselves to pairs (m, r) when signing. In fact, the construction of Scheme

B can be generalized to obtain Scheme C which can sign tuples $(m^{(0)}, \ldots, m^{(\ell)})$, i.e., blocks of messages.

Scheme C consists of the following algorithms:

Key generation. Run the *Setup* algorithm to generate $(q, G, \mathsf{G}, g, \mathsf{g}, e)$. Choose $x \leftarrow \mathbb{Z}_q, y \leftarrow \mathbb{Z}_q$, and for $1 \leq i \leq \ell$, $z_i \leftarrow \mathbb{Z}_q$. Let $X = g^x, Y = g^y$ and, for $1 \leq i \leq \ell$, $Z_i = g^{z_i}$. Set $sk = (x, y, z_1, \ldots, z_\ell)$, $pk = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y, \{Z_i\})$.

Signature. On input message $(m^{(0)}, m^{(1)}, \ldots, m^{(\ell)})$, secret key $sk = (x, y, z_1, \ldots, z_\ell)$, and public key $pk = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y, \{Z_i\})$ do:

- Choose a random $a \leftarrow G$.
- Let $A_i = a^{z_i}$ for $1 \leq i \leq \ell$.
- Let $b = a^y$, $B_i = (A_i)^y$.
- Let $c = a^{x+xym^{(0)}} \prod_{i=1}^{\ell} A_i^{xym^{(i)}}$.

Output $\sigma = (a, \{A_i\}, b, \{B_i\}, c)$.

Verification. On input $pk = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y, \{Z_i\})$, message $(m^{(0)}, \ldots, m^{(\ell)})$, and purported signature $\sigma = (a, \{A_i\}, b, \{B_i\}, c)$, check the following:

1. $\{A_i\}$ were formed correctly: $e(a, Z_i) = e(g, A_i)$.
2. b and $\{B_i\}$ were formed correctly: $e(a, Y) = e(g, b)$ and $e(A_i, Y) = e(g, B_i)$.
3. c was formed correctly: $e(X, a) \cdot e(X, b)^{m^{(0)}} \cdot \prod_{i=1}^{\ell} e(X, B_i)^{m^{(i)}} = e(g, c)$.

The proof that this scheme is secure and correct is deferred to Corollary 1.

4 Anonymous Credential System and Group Signature Scheme

Following Camenisch and Lysyanskaya [10,29], in order to construct an anonymous credential system, it is sufficient to exhibit a commitment scheme, a signature scheme, and efficient protocols for (1) proving equality of two committed values; (2) getting a signature on a committed value (without revealing this value to the signer); and (3) proving knowledge of a signature on a committed value. We provide all these tools in this section.

Constructing a group signatures scheme or identity escrow scheme additionally requires an encryption scheme that is secure against adaptively chosen ciphertext attacks and a protocol that a committed value is contained in a ciphertext (cf. [12,3,11]). Camenisch and Shoup provide an encryption scheme and such a protocol [11]. However, in our case we could also use the Cramer-Shoup encryption scheme [18], provided that the order of the group over which encryption is carried out is the same as the order of the group over which our signature scheme is constructed. This will allow for a more efficient proof that a ciphertext contains information to identify a group member and thus a more efficient group signatures/identity escrow scheme. We will describe the details of this in §4.4.

The reason that our new signature schemes are particularly suitable for the credential scheme application, is the fact that, given one signature on a given message, it is easy to generate another one. Consider Signature Scheme A. From a signature $\sigma = (a, b, c)$ on message m, it is very easy to compute a different signature $\sigma = (\tilde{a}, \tilde{b}, \tilde{c})$ on the same message m: just choose a random $r \in \mathbb{Z}_q$ and let $\tilde{a} = a^r$, $\tilde{b} = b^r$, $\tilde{c} = c^r$. This alone is, of course, not sufficient, but this already shows the way in which the pieces of our credential scheme will fall into place.

4.1 The Relevant Commitment Scheme

Recall the Pedersen commitment scheme [33]: given a group G of prime order q with generators g and h, a commitment to $x \in \mathbb{Z}_q$ is formed by choosing a random $r \leftarrow \mathbb{Z}_q$ and setting the commitment $C = g^x h^r$. This commitment scheme is information-theoretically hiding, and is binding under the discrete logarithm assumption, which is implied by the LRSW assumption. Moreover, there exist in the literature efficient protocols for proving knowledge and equality of committed values (see, for example, [16,36,8,15]).

4.2 Obtaining a Signature on a Committed Value

When Information-Theoretic Hiding Is not Needed. Consider the signing algorithm for Scheme A. Note that, if the input to the signer is g^m instead of m, the algorithm will still work: on input $M = g^m$, output $a = g^r$, $b = a^y$, and $c = a^x M^{rxy} = a^{x+mxy}$. To maintain security of the signature scheme, however, the user must prove knowledge of m to the signer.

As we will discuss in more detail in §4.4, this leads to a natural application to constructing group signatures: in order to join a group, a new member will choose a secret m, give g^m to the group manager, prove knowledge of m, and obtain the membership certificate (a, b, c) formed as above.

However, note here that the input to the signer, the value g^m, does not unconditionally hide the value m. Thus, if the user wishes to become a member in more than one group using the same secret m (as is the case if we want to build an anonymous credential system), the two group managers can discover that they are talking to the same user. This is easy to see if both group managers use the same generator g for G, because in that case, the user will give g^m to both of them. But this is true even if one group manager uses g, while the other uses $\tilde{g}$: recall that in groups with bilinear pairings, the decisional Diffie-Hellman problem is easy, and so g^m and $\tilde{g}^m$ can be correlated: $e(g^m, \tilde{g}) = e(g, \tilde{g})^m = e(g, \tilde{g}^m)$.

This is why we need Schemes B and C instead of Scheme A. However, we note that for group signatures, Scheme A is sufficient. In the sequel, we will give the description of the protocol for Scheme C, together with a proof of security. Because Scheme A is a special case of Scheme C (in Scheme A, $\ell = 0$), the security of the protocols for A is implied by that for C.

Signing an Information-Theoretically Hidden Message. Signature Schemes B and C are ideally suited for obtaining a signature on a committed value.

Consider Signature Scheme B. Note that to generate a valid signature, the signer need not know (m, r). Instead, it is sufficient that the signer know $M =$

$g^m Z^r$. The values (a, A, b, B) are not a function of (m, r) — so the signer need not know (m, r) to generate them. Suppose that the signer generates them as follows: choose $\alpha \leftarrow \mathbb{Z}_q$, and let $a = g^\alpha$. Choose A, b, and B as prescribed by the signing algorithm. Finally, the signer can compute $c = a^{x+xym}A^{xyr}$ as $c = a^x M^{\alpha xy}$. This will be correct, because:

$$\begin{aligned} c &= a^x M^{\alpha xy} \\ &= a^x (g^m Z^r)^{\alpha xy} \\ &= a^x (a^m A^r)^{xy} && \text{because by construction, } A = g^{\alpha z} = Z^\alpha \\ &= a^{x+xym}A^{xyr} \end{aligned}$$

More generally, in Signature Scheme C, all the signer needs is the value $M = g^{m^{(0)}} \prod_{i=1}^{\ell} Z_i^{m^{(i)}}$. He can then compute $(a = g^\alpha, \{A_i\}, b, \{B_i\})$ as prescribed, and let $c = a^x M^{\alpha xy}$ as above.

We do not know how to prove such a method for signing secure under the LRSW assumption: the difference from the usual method is that here, the adversary may win by asking a signature query for M for which he does not know the representation in terms of g and Z.

Thus, in order to obtain a signature on a committed value, the protocol needs to be amended by having a recipient of the signature prove that he knows the representation of M in bases g and Z.

Let us give the protocol in detail now. We give the protocol for Signature Scheme C, the ones for Signature Schemes A and B follow from this as they are special cases of Signature Scheme C.

Obtaining a Signature C on a Committed Value. Suppose that $M = g^{m^{(0)}} \prod_{i=1}^{\ell} Z_i^{m^{(i)}}$ is a commitment to a set of messages $(m^{(0)}, \ldots, m^{(\ell)})$ whose signature the user wishes to obtain. Then the user and the signer run the following protocol:

Common Input. The public key $pk = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y, \{Z_i\})$, and a commitment M.

User's Input. Values $m^{(0)}, \ldots, m^{(\ell)}$ such that $M = g^{m^{(0)}} \prod_{i=1}^{\ell} Z_i^{m^{(i)}}$.

Signer's Input. Signing key $sk = (x, y, \{z_i\})$.

Protocol. First, the user gives a zero-knowledge proof of knowledge of the opening of the commitment:

$$PK\{(\mu^{(0)}, \ldots, \mu^{(\ell)}) : M = g^{\mu^{(0)}} \prod_{i=1}^{\ell} Z_i^{\mu^{(i)}}\}$$

Next, the signer computes $\sigma = (a, \{A_i\}, b, \{B_i\}, c)$ as described above, namely:

- $\alpha \leftarrow \mathbb{Z}_q$, $a = g^\alpha$.
- For $1 \leq i \leq \ell$, let $A_i = a^{z_i}$. Then set $b = a^y$, and for $1 \leq i \leq \ell$, let $B_i = A_i^y$.

– $c = a^x M^{\alpha x y}$.

The user outputs the signature σ.

Theorem 3. *The protocol above is a secure two-party computation of a signature on a discrete-logarithm representation of M under the signer's public key.*

Proof. (Sketch) From the signer's point of view, this protocol is as secure as when the user submits his signature queries in the clear. This is because of the proof of knowledge: there exists an extractor that can discover the value of the message being signed, and ask it of the signer in the clear.

From the user's point of view, as the only place where the user's secret input $(m^{(0)}, \ldots, m^{(\ell)})$ is used is the zero-knowledge proof of knowledge of these values, the only thing that the signer finds out about the message $(m^{(0)}, \ldots, m^{(\ell)})$, is the input value M. Note that if $m^{(\ell)}$ is distributed uniformly at random, then M information-theoretically hides the values $(m^{(0)}, \ldots, m^{(\ell-1)})$.

4.3 Proving Knowledge of a Signature

We first present a protocol to prove knowledge of a signature that works for Scheme A. We then explain why the protocol does not generalize to Scheme B (and thus also Scheme C), show how Scheme C needs to be extended to fix this problem, and obtain Scheme D. We then give a proof of security of Scheme D and a zero-knowledge protocol for proving knowledge of a signature under Scheme D. We note that the protocol to sign a committed (secret) message also works for Scheme D.

The following protocol is a zero-knowledge proof of knowledge of a signed message for Scheme A.

Common input. The public key $pk = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y)$.
Prover's input. The message $m \in \mathbb{Z}_q$ and signature $\sigma = (a, b, c)$.
Protocol. The prover does the following:

1. Compute a blinded version of his signature σ: Choose random $r, r' \in \mathbb{Z}_q$, and blind the signature to form $\tilde{\sigma} := (a^{r'}, b^{r'}, c^{r'r}) = (\tilde{a}, \tilde{b}, \tilde{c}^r) = (\tilde{a}, \tilde{b}, \hat{c})$. Send $(\tilde{a}, \tilde{b}, \hat{c})$ to the verifier.
2. Let the v_x, v_{xy}, and v_s be as follows:

$$\mathsf{v}_x = e(X, \tilde{a}) \ , \qquad \mathsf{v}_{xy} = e(X, \tilde{b}) \ , \qquad \mathsf{v}_s = e(g, \hat{c}) \ .$$

The Prover and Verifier compute these values (locally) and then carry out the following zero-knowledge proof protocol:

$$PK\{(\mu, \rho) : \mathsf{v}_s^{\rho} = \mathsf{v}_x \mathsf{v}_{xy}^{\mu}\} \ .$$

The Verifier accepts if it accepts the proof above and $e(\tilde{a}, Y) = e(g, \tilde{b})$.

Theorem 4. *The protocol above is a zero knowledge proof of knowledge of a signature σ on a message m under Signature Scheme A.*

Proof. First, we prove the zero-knowledge property. The values that the verifier receives from the prover in Step 1 are independent of the actual signature: $\tilde{a}$ and $\tilde{b}$ are just random values satisfying $e(\tilde{a}, Y) = e(g, \tilde{b})$, and $\hat{c}$ is random in G because $\hat{c} = \tilde{c}^{r'}$ for a randomly chosen r'. Therefore, consider the following simulator S: Choose random r and r', and set $\tilde{a} = g^r$, $\tilde{b} = Y^r$, $\hat{c} = g^{r'}$. Then $(\tilde{a}, \tilde{b}, \hat{c})$ is distributed correctly, and so Step 1 is simulated correctly. Then, because in Step 2, the Prover and Verifier execute a zero-knowledge proof, it follows that there exists a simulator S' for this step; just run S'. It is easy to see that S constructed this way is the zero-knowledge simulator for this protocol.

Next, let us prove that this protocol is a proof of knowledge. That is to say, we must exhibit a knowledge extractor algorithm E that, given access to a Prover such that the Verifier's acceptance probability is non-negligible, outputs a value (m, σ), such that σ is a valid signature. Suppose that we are given such a prover. The extractor proceeds as follows: first, it runs the extractor for the proof of knowledge protocol of Step 2. As a result, it obtains the values $r, m \in \mathbb{Z}_q$ such that $\mathsf{v}_s^r = \mathsf{v}_x \mathsf{v}_{xy}^m$. Then:

$$\begin{aligned} \mathsf{v}_s^r &= \mathsf{v}_x \mathsf{v}_{xy}^m \\ e(g, \hat{c})^r &= e(X, \tilde{a}) e(X, \tilde{b})^m \\ e(g, \hat{c}^r) &= e(X, \tilde{a}) e(X, \tilde{b})^m \end{aligned}$$

And therefore the triple $\sigma = (\tilde{a}, \tilde{b}, \hat{c}^r)$ satisfies the verification equation (1) and hence is a signature on the message m, so our extractor outputs (m, σ).

Let us now try to adapt this protocol for Signature Scheme C. There is one subtlety that arises here: The zero-knowledge simulator needs to be able to come up with something that looks like a blinded signature (let us call it *simulated signature*), even though *the simulator is not given any signature.* In Signature Scheme A this turned out not to be a problem: the simulator simply picked a random r and set $\tilde{a} = g^r$, and $\tilde{b} = Y^r$. Here, this is not going to work, because, in addition to $\tilde{a}$ and $\tilde{b}$, the simulated signature needs to include the values $\{\tilde{A}_i\}$ and $\{\tilde{B}_i\}$. Now, forming $\tilde{A}_i$ is not a problem: $\tilde{A}_i = Z_i^r$. But how do we compute $\tilde{B}_i = \tilde{A}_i^y = g^{r z_i y}$ without knowing z_i or y?

To that end, we may augment the public key for signature scheme C to include a signature on some dummy message, so that the simulator will be given *some* valid signature that includes the correctly formed tuple $(a, \{A_i\}, b, \{B_i\})$, and then, in order to obtain the simulated signature, the simulator will pick a random r, and let $\tilde{a} = a^r$, $\tilde{b} = b^r$, $\tilde{A}_i = A_i^r$, and $\tilde{B}_i = B_i^r$.

An even better solution, in terms of reducing the size of the public key, is actually to include the values $W_i = Y^{z_i}$ in the public key, instead of the signature on the dummy message. It is easy to see that this has no effect on the security of the signature scheme.

Let us now give this new, augmented signature scheme, and prove it secure.

Signature Scheme D. This signature scheme is *the same as Signature Scheme C*, except that the public key also includes the values $\{W_i = Y^{z_i}\}$.

Key generation. Run the *Setup* algorithm to generate $(q, G, \mathsf{G}, g, \mathsf{g}, e)$. Choose $x \leftarrow \mathbb{Z}_q$, $y \leftarrow \mathbb{Z}_q$, and for $1 \leq i \leq \ell$, $z_i \leftarrow \mathbb{Z}_q$. Let $X = g^x$, $Y = g^y$ and, for $1 \leq i \leq \ell$, $Z_i = g^{z_i}$ and $W_i = Y^{z_i}$. Set $sk = (x, y, z_1, \ldots, z_\ell)$, $pk = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y, \{Z_i\}, \{W_i\})$.

The signature and verification algorithm are identical to the ones of Scheme C.

Theorem 5. *Signature Scheme D is correct and secure under the LRSW assumption.*

The detailed proof of this theorem is given in the full version of this paper. The main idea of the proof of security is that the proof for Scheme B generalizes to the case when we have several Z_i's.

As a forger for Scheme C is also a forger for Scheme D, we have:

Corollary 1. *Signature Scheme C is correct and secure under the LRSW assumption.*

The full description of the protocol and proof of security follow.

Common input. The public key $pk = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y, \{Z_i\}, \{W_i\})$.
Prover's input. The block of messages $(m^{(0)}, \ldots, m^{(\ell)})$ and signature $\sigma = (a, \{A_i\}, b, \{B_i\}, c)$.
Protocol. The prover does the following:

1. Compute a blinded version of his signature σ: Choose random $r, r' \in \mathbb{Z}_q$. Form $\tilde{\sigma} = (\tilde{a}, \{\tilde{A}_i\}, \tilde{b}, \{\tilde{B}_i\}, \tilde{c})$ as follows:
$$\tilde{a} = a^r, \quad \tilde{b} = b^r \quad \text{and} \quad \tilde{c} = c^r$$
$$\tilde{A}_i = A_i^r \quad \text{and} \quad \tilde{B}_i = B_i^r \text{ for } 1 \leq i \leq \ell$$
Further, blind $\tilde{c}$ to obtain a value $\hat{c}$ that it is distributed independently of everything else: $\hat{c} = \tilde{c}^{r'}$.
Send $(\tilde{a}, \{\tilde{A}_i\}, \tilde{b}, \{\tilde{B}_i\}, \hat{c})$ to the verifier.
2. Let v_x, v_{xy}, $\mathsf{V}_{(xy,i)}$, $i = 1, \ldots, \ell$, and v_s be as follows:
$$\mathsf{v}_x = e(X, \tilde{a}) \;\;,\;\; \mathsf{v}_{xy} = e(X, \tilde{b}) \;\;,\;\; \mathsf{V}_{(xy,i)} = e(X, \tilde{B}_i) \;\;,\;\; \mathsf{v}_s = e(g, \hat{c})$$
The Prover and Verifier compute these values (locally) and then carry out the following zero-knowledge proof protocol:
$$PK\{(\mu^{(0)}, \ldots, \mu^{(\ell)}, \rho) : (\mathsf{v}_s)^\rho = \mathsf{v}_x (\mathsf{v}_{xy})^{\mu^{(0)}} \prod_{i=1}^{\ell} (\mathsf{V}_{(xy,i)})^{\mu^{(i)}}\}$$
The Verifier accepts if it accepts the proof above and (a) $\{\tilde{A}_i\}$ were formed correctly: $e(\tilde{a}, Z_i) = e(g, \tilde{A}_i)$; and (b) $\tilde{b}$ and $\{\tilde{B}_i\}$ were formed correctly: $e(\tilde{a}, Y) = e(g, \tilde{b})$ and $e(\tilde{A}_i, Y) = e(g, \tilde{B}_i)$.

Theorem 6. *The protocol above is a zero knowledge proof of knowledge of a signature σ on a block of messages $(m^{(0)}, \ldots, m^{(\ell)})$ under Signature Scheme D.*

The proof of this theorem follows the proof of Theorem 4 and is provided in the full version of this paper.

4.4 An Efficient Group Signature Scheme Secure under the LSWR-Assumption

We now present the first efficient group signature (and identity escrow) scheme whose security relies solely on assumptions related to the discrete logarithm problem (in the random oracle model). In contrast, all previous efficient schemes rely on the strong RSA assumption plus the decisional Diffie-Hellman assumption.

Recall that a group signatures scheme allows members of a group to sign anonymously on the group's behalf. In case of disputes, there exists a trusted third party called revocation manager who will be able to open a signature and reveal the identity of the signer. A group signature scheme consists of five procedures: (1) a key generation procedure that produces the public key of the group (and also some keys for the group and revocation manager), (2) a join protocol for a member to get admitted by the group manager, (3) a sign algorithm for an admitted member to sign a message, (4) a verification algorithm to check group signatures for validity with respect to the group's public key, and (5) an opening algorithm that allows the revocation manager to reveal the identity of a signer. A group signature scheme is secure if only the revocation manager can reveal the identity of the signer (anonymity) and if the revocation manager can do this for all valid signatures (traceability) [3].

Our construction follows the approach introduced by Camenisch and Stadler [12]: A member gets a certificate on a membership public key from the group manager when she joins the group. When she wants to sign on behalf of the group, she encrypts her membership public key under the encryption key of the party who will later be able to open group signatures (revocation manager) and then proves that she possesses a certificate on the encrypted membership public key and that she knows its secret key. To make this proof a signature, one usually applies the Fiat-Shamir heuristic to this proof [21].

The public key of the group manager is the public key of our Scheme A, i.e., $pk_M = (q, G, \mathsf{G}, g, \mathsf{g}, e, X, Y)$ and his secret key is $x = \log_g X$ and $y = \log_g Y$. The public key of the revocation manager is the public key of the Cramer-Shoup encryption scheme [18] in the group $\mathsf{G} = \langle \mathsf{g} \rangle$, i.e., $pk_R = (\mathsf{h}, \mathsf{y}_1, \mathsf{y}_2, \mathsf{y}_3)$, with $\mathsf{h} \in_R \mathsf{G}$, $\mathsf{y}_1 = \mathsf{g}^{x_1}\mathsf{h}^{x_2}$, $\mathsf{y}_2 = \mathsf{g}^{x_3}\mathsf{h}^{x_4}$, and $\mathsf{y}_3 = \mathsf{g}^{x_5}$, where $x_1, \ldots, x_5 \in_R \mathbb{Z}_q$ are the revocation manager's secret key.[1] Finally, let $\mathcal{H}() : \{0,1\}^* \rightarrow \mathbb{Z}_q$ be a collision resistant hash function (modeled as a random oracle in the proof of security).

The join protocol is as follows. The future group member chooses her membership secret key $k \in_R \mathbb{Z}_q$, sets $P = g^k$, sends P authentically to the group manager, and proves to the group manager the knowledge of $\log_g P$. The group manager replies with a Scheme A signature (a, b, c) on the message committed by P, i.e., computes $a = g^r$, $b = a^y$, and $c = a^x P^{rxy}$, where $r \in_R \mathbb{Z}_q$ (cf. §4.2).

[1] The Cramer-Shoup cryptosystem is secure under the decisional Diffie-Hellman (DDH) assumption. Therefore, we cannot use it over group G, because the existence of a bilinear map implies that the DDH problem is tractable. Thus, we use the CS cryptosystem in group G instead.

The group manager stores $\mathsf{P} = e(P, g)$ together with P and the identity of the new group member.

To sign a message m on behalf of the group, the user computes $\mathsf{P} = \mathsf{g}^k = e(P, g)$ and a blinded version of the certificate by choosing random $r, r' \in \mathbb{Z}_q$ and computing $\tilde{\sigma} := (a^{r'}, b^{r'}, c^{r'r}) = (\tilde{a}, \tilde{b}, \tilde{c}^r) = (\tilde{a}, \tilde{b}, \hat{c})$. Next, she encrypts P under the revocation manager's public key pk_R, i.e., she chooses $u \in_R \mathbb{Z}_q$, computes $\mathsf{c}_1 = \mathsf{g}^u$, $\mathsf{c}_2 = \mathsf{h}^u$, $\mathsf{c}_3 = \mathsf{y}_1^u \mathsf{P}$, and $\mathsf{c}_4 = \mathsf{y}_2^u \mathsf{y}_3^{u\mathcal{H}(\mathsf{c}_1 \| \mathsf{c}_2 \| \mathsf{c}_3)}$. Then she computes the following proof-signature (cf. §2):

$$\begin{aligned} \Sigma = SPK\{(\mu, \rho, \upsilon) : \mathsf{v}_s^\rho = \mathsf{v}_x \mathsf{v}_{xy}^\mu \ \wedge\ \mathsf{c}_1 = \mathsf{g}^\upsilon \ \wedge\ \mathsf{c}_2 = \mathsf{h}^\upsilon \ \wedge \\ \wedge\ \mathsf{c}_3 = \mathsf{y}_1^\upsilon \mathsf{g}^\mu \ \wedge\ \mathsf{c}_4 = (\mathsf{y}_2 \mathsf{y}_3^{\mathcal{H}(\mathsf{c}_1 \| \mathsf{c}_2 \| \mathsf{c}_3)})^\upsilon\}(m) \ , \end{aligned}$$

where $\mathsf{v}_x = e(X, \tilde{a})$, $\mathsf{v}_{xy} = e(X, \tilde{b})$, and $\mathsf{v}_s = e(g, \hat{c})$. A group signature consists of $((\tilde{a}, \tilde{b}, \hat{c}), (\mathsf{c}_1, \mathsf{c}_2, \mathsf{c}_3, \mathsf{c}_4), \Sigma)$ and is valid if Σ is a valid *SPK* as defined above and if $e(\tilde{a}, Y) = e(g, \tilde{b})$ holds.

To open such a group signature, the revocation managers needs to decrypt $(\mathsf{c}_1, \mathsf{c}_2, \mathsf{c}_3, \mathsf{c}_4)$ to obtain P which identifies the group member.

It is not hard to see that, in the random oracle model, this is a secure group signatures scheme under the LRSW and the decisional Diffie-Hellman assumption in G. Let us give a proof sketch for security under the Bellare et al. [3] definition. If an adversary can break anonymity, then one can break the encryption scheme as $(\tilde{a}, \tilde{b}, \hat{c})$ are random values and Σ is derived from an honest-verifier zero-knowledge proof. If an adversary can produce a signature that cannot be opened, i.e., linked to a registered member by the revocation manager, then one can use rewinding to extract a forged signature and break the signature scheme (cf. analysis of the protocol to prove knowledge of a signatures in §4.3). If used as an identity escrow scheme (i.e., if Σ is not a proof-signature but a real protocol between a group member and a verifier), the security proof need not to assume random oracles.

The scheme just described can be extended in several ways. For instance, we could use Scheme D instead of Scheme A and include the user's identity `id` directly into her membership key P, e.g., $P = g^k Z_1^{\mathtt{id}}$. That is, in the join protocol, the user would send $P' = g^k$ (and prove knowledge of $\log_g P$) and the group manager would then compute P as to ensure that indeed `id` is contained in P. Then, instead of encrypting P, one could use the Camenisch-Shoup encryption scheme [11] to directly encrypt the identity as one of the discrete logarithms the knowledge of which is proven when proving knowledge of a signature.

5 Constructions Based on the BBS Group Signature

Recently and independently of this work, Boneh, Boyen and Shacham [5] presented a group signature scheme secure under the strong Diffie-Hellman and the Linear assumptions. They showed that, under these assumptions in groups with bilinear pairings, it is hard, on input $(g_1, g_2 = g_1^\gamma)$ to sample tuples of the form

(A, x) where $A = g_1^{1/(\gamma+x)}$ (in other words, $A^{\gamma+x} = g_1$), even given a polynomial number of such samples. In their group signature scheme, such a tuple (A, x) is a user's group membership certificate, while (g_1, g_2) is the public key of the group. At the heart of their construction are (1) a zero-knowledge proof of knowledge of such a tuple; and (2) a scheme for encrypting x. They prove the resulting construction secure under a slightly weaker variant of the Bellare, Micciancio, and Warinschi [3] definition of security.

Boneh, Boyen, and Shacham also modify their main group signature scheme to achieve exculpability, as follows. The public key of the group is augmented by an additional value h; it is now (g_1, g_2, h). The membership certificate of a group member is (A, x, y) such that $A^{\gamma+x}h^y = g_1$. This membership certificate is created via a protocol in which the group manager only learns the value h^y, but not the value y. The unforgeability of membership certificates in this modified scheme can be derived from that of their main scheme. They achieve exculpability because a proof of knowledge of a membership certificate requires the knowledge of the value y.

Note that this latter signature scheme gives rise to the equivalent of our Signature Scheme A, but under a different assumption. Namely, the membership certificate (A, x, y) is a signature on the value y. Just as in our Scheme A, a group member obtains his group membership certificate in such a way that the group manager learns the value h^y but not the value y itself.

Not surprisingly, this signature scheme can be extended to the equivalent of our Schemes B and C using techniques similar to the ones described above. As a result, we can obtain signature schemes with efficient protocols based on the BBS signature. Let us give a sketch for the equivalent for Scheme C. A public key would be $(g_1, g_2, h_0, h_1, \ldots, h_\ell)$. A signature on a block of messages $(m_0, \ldots, m_\ell)$ consists of values (A, x) such that $A^{\gamma+x} \prod_{i=0}^{\ell} h_i^{m_i}$. In order to obtain a signature on a committed block of messages, a user will have to supply the signer with the value $Y = \prod_{i=0}^{\ell} h_i^{m_i}$, and prove knowledge of its representation in the bases $(h_0, \ldots, h_\ell)$. If m_0 is chosen at random, then Y information-theoretically hides $(m_1, \ldots, m_\ell)$. The signer will then generate the signature. A proof of knowledge of a signature on a committed value can be obtained by appropriate modifications to the BBS group signature protocol.

Acknowledgments

We thank Dan Boneh, Xavier Boyen and Hovav Shacham for making their paper [5] available to us as we were preparing the final version of this paper. We also thank the anonymous referees for helpful comments. Anna Lysyanskaya is supported by NSF Career grant CNS-0347661.

References

1. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *CRYPTO 2000*, vol. 1880 of *LNCS*, pp. 255–270. Springer Verlag, 2000.

2. G. Ateniese and B. de Medeiros. Efficient group signatures without trapdoors. In *ASIACRYPT 2003*, vol. 2894 of *LNCS*, pp. 246–268. Springer Verlag, 2003.
3. M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: Formal definition, simplified requirements and a construction based on general assumptions. In *Eurocrypt 2003*, vol. 2656 of *LNCS*, pp. 614–629, 2003.
4. D. Boneh and X. Boyen. Short signatures without random oracles. In *EUROCRYPT 2004*. Springer Verlag, 2004.
5. D. Boneh, X. Boyen, and H. Shacham. Short group signatures using strong diffie hellman. In *CRYPTO 2004*. Springer Verlag, 2004.
6. D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. In *CRYPTO 2001*, vol. 2139 of *LNCS*, pp. 213–229. Springer Verlag, 2001.
7. D. Boneh and A. Silverberg. Applications of multilinear forms to cryptography. In *Topics in Algebraic and Noncommutative Geometry, Contemporary Mathematics*, vol. 324, pp. 71–90. American Mathematical Society, 2003.
8. S. Brands. Rapid demonstration of linear relations connected by boolean operators. In *EUROCRYPT '97*, vol. 1233 of *LNCS*, pp. 318–333. Springer Verlag, 1997.
9. J. Camenisch and A. Lysyanskaya. Efficient non-transferable anonymous multi-show credential system with optional anonymity revocation. In *EUROCRYPT 2001*, vol. 2045 of *LNCS*, pp. 93–118. Springer Verlag, 2001.
10. J. Camenisch and A. Lysyanskaya. A signature scheme with efficient protocols. In *Security in communication networks*, vol. 2576 of *LNCS*, pp. 268–289, 2002.

CL04. Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. `http://eprint.iacr.org`, 2004.

11. J. Camenisch and V. Shoup. Practical verifiable encryption and decryption of discrete logarithms. In *CRYPTO 2003*, *LNCS*, pp. 126–144. Springer Verlag, 2003.
12. J. Camenisch and M. Stadler. Efficient group signature schemes for large groups. In *CRYPTO '97*, vol. 1296 of *LNCS*, pp. 410–424. Springer Verlag, 1997.
13. R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. In *Proc. 30th Annual ACM STOC*, pp. 209–218, 1998.
14. D. Chaum. Security without identification: Transaction systems to make big brother obsolete. *Communications of the ACM*, 28(10):1030–1044, Oct. 1985.
15. D. Chaum, J.-H. Evertse, and J. van de Graaf. An improved protocol for demonstrating possession of discrete logarithms and some generalizations. In *EUROCRYPT '87*, vol. 304 of *LNCS*, pp. 127–141. Springer-Verlag, 1988.
16. D. Chaum and T. P. Pedersen. Wallet databases with observers. In *CRYPTO '92*, vol. 740 of *LNCS*, pp. 89–105. Springer-Verlag, 1993.
17. D. Chaum and E. van Heyst. Group signatures. In *EUROCRYPT '91*, vol. 547 of *LNCS*, pp. 257–265. Springer-Verlag, 1991.
18. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *CRYPTO '98*, vol. 1642 of *LNCS*, pp. 13–25, Berlin, 1998. Springer Verlag.
19. R. Cramer and V. Shoup. Signature schemes based on the strong RSA assumption. In *Proc. 6th ACM CCS*, pp. 46–52. ACM press, nov 1999.
20. W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Trans. on Information Theory*, IT-22(6):644–654, Nov. 1976.
21. A. Fiat and A. Shamir. How to prove yourself: Practical solution to identification and signature problems. In *CRYPTO '86*, vol. 263 of *LNCS*, pp. 186–194, 1987.
22. M. Fischlin. The Cramer-Shoup strong-RSA signature scheme revisited. In *Public Key Cryptography - PKC 2003*, vol. 2567 of *LNCS*. Springer-Verlag, 2002.
23. R. Gennaro, S. Halevi, and T. Rabin. Secure hash-and-sign signatures without the random oracle. In *EUROCRYPT '99*, vol. 1592 of *LNCS*, pp. 123–139, 1999.

24. C. Gentry and A. Silverberg. Hierarchical ID-based cryptography. In *ASIACRYPT 2002*, vol. 2501 of *LNCS*, pp. 548–566. Springer Verlag, 2002.
25. S. Goldwasser and Y. T. Kalai. On the (in)security of the Fiat-Shamir paradigm. In *Proc. 44th IEEE FOCS*, pp. 102–115. IEEE Computer Society Press, 2003.
26. S. Goldwasser, S. Micali, and R. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, Apr. 1988.
27. A. Joux. A one-round protocol for tripartite Diffie-Hellman. In *Proceedings of the ANTS-IV conference*, vol. 1838 of *LNCS*, pp. 385–394. Springer-Verlag, 2000.
28. J. Kilian and E. Petrank. Identity escrow. In *CRYPTO '98*, vol. 1642 of *LNCS*, pp. 169–185, Berlin, 1998. Springer Verlag.
29. A. Lysyanskaya. *Signature Schemes and Applications to Cryptographic Protocol Design*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, Sept. 2002.
30. A. Lysyanskaya, R. Rivest, A. Sahai, and S. Wolf. Pseudonym systems. In *Selected Areas in Cryptography*, vol. 1758 of *LNCS*. Springer Verlag, 1999.
31. S. Micali. 6.875: Introduction to cryptography. MIT course taught in Fall 1997.
32. M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *Proc. 21st Annual ACM STOC*, pp. 33–43, 1989. ACM.
33. T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO '91*, vol. 576 of *LNCS*, pp. 129–140. Springer Verlag, 1992.
34. R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.
35. J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proc. 22nd Annual ACM STOC*, pp. 387–394, Baltimore, Maryland, 1990. ACM.
36. C. P. Schnorr. Efficient signature generation for smart cards. *Journal of Cryptology*, 4(3):239–252, 1991.
37. J. Silverman. *The Arithmetic of Elliptic Curves*. Springer-Verlag, 1986.
38. E. Verheul. Self-blindable credential certificates from the weil pairing. In *ASIACRYPT 2001*, vol. 2248 of *LNCS*, pp. 533–551. Springer Verlag, 2001.