Group Blind Digital Signatures: A Scalable Solution to Electronic Cash

Anna Lysyanskaya[1] and Zulfikar Ramzan[1]

Laboratory for Computer Science,
Massachusetts Institute of Technology,
Cambridge MA 02139,
{anna, zulfikar}@theory.lcs.mit.edu

Abstract. In this paper we construct a practical group blind signature scheme. Our scheme combines the already existing notions of blind signatures and group signatures. It is an extension of Camenisch and Stadler's Group Signature Scheme [5] that adds the blindness property. We show how to use our group blind signatures to construct an electronic cash system in which multiple banks can securely distribute anonymous and untraceable e-cash. Moreover, the identity of the e-cash issuing bank is concealed, which is conceptually novel. The space, time, and communication complexities of the relevant parameters and operations are independent of the group size.

1 Introduction

1.1 Distributed Electronic Banking

Consider a scheme in which there is a large group of banks, monitored by the country's Central Bank (e.g. the US Treasury), where each bank can dispense electronic cash. We want such a scheme to have the following properties:

1. No bank should be able to trace any e-cash it issues. Therefore, just as with paper money, people can spend their e-cash anonymously.
2. A vendor only needs to invoke a single universal verification procedure, based on the group public key, to ensure the validity of any e-cash he receives. This procedure works regardless of which bank issued the e-cash. This makes the vendor's task much easier since he only needs to know the single group public key.
3. There is a single public key for the entire group of banks. The size of this public key is independent of the number of banks. Moreover, the public key should not be modified if more banks join the group. Thus, the scheme is still practical even if there are a large number of participating banks.
4. Given a valid piece of e-cash only the Central Bank can tell which bank in the group issued it. No vendor can even determine the bank from which the customer got her e-cash even though the vendor can easily check that the e-cash is valid. This restriction gives an extra layer of anonymity since we conceal both the spender's identity and the bank she uses.

5. Neither the Central Bank nor any bank can issue cash on behalf of another bank; i.e. no bank or any other entity, including the Central Bank can "frame" another bank.

In this paper we implement such a scheme using Group Blind Digital Signatures. Many previous electronic cash schemes focus on a model in which a single bank distributes all the e-cash. In real life, one would like for more than one bank to be able to dispense electronic money. Our scheme is unique since it considers scalability as a criterion in the design of electronic cash systems. Moreover, our scheme is conceptually novel since we conceal the bank's identity in addition to the spender's.

1.2 Blind Digital Signatures

Blind Digital Signatures were introduced by Chaum [7] to allow for spender anonymity in Electronic Cash systems. Such signatures require that a signer be able to sign a document without knowing its contents. Moreover, should the signer ever see the document/signature pair, he should have no recollection of it (even though he can verify that the signature is indeed his). In the electronic cash scenario, a document corresponds to an electronic coin, and the signer represents a bank. The spender retains anonymity in any transaction that involves electronic coins since they are blindly signed.

Blind Signatures have been looked at extensively in the literature [6, 16, 14]. Like some of the previous schemes, the security of our scheme is proven under the random oracle model [16, 1, 11]. Complexity based proofs of security are, however, considered much stronger. Unfortunately, the only known Blind Signature Scheme with such a proof is far too impractical [14].

1.3 Group Digital Signatures

In a group digital signature scheme, members of a given group are allowed to sign on behalf of the entire group. In addition, the signatures can be verified using a single *group public key*. Also, once a document is signed, no one, except a designated group manager, can determine which member of the group signed it. Companies can use group signatures to validate price lists, press releases or digital contracts; customers would only need to know a single company public key to verify signatures. Companies can then conceal their internal structure, but can still determine which employee signed a given document. Group Digital Signatures were first introduced and implemented by Chaum and van Heyst [9]. Recently, Camenisch and Stadler [5] presented the first group signature scheme for which the size of the group public key remains independent of the group size, as do the time, space, and, communication complexities of the necessary operations; in previous schemes [9, 3, 10] the size of the public key grew with the size of the group which is impractical for large groups. We extend [5] by showing how to add the blindness property to their scheme. This may have applications besides the electronic cash and voting schemes described below.

1.4 Using Group Blind Signatures for Electronic Cash

If we combine the properties of Blind Signatures and Group Signatures, we get what we call *Group Blind Digital Signatures*. We now show how to use group blind digital signatures to achieve a scheme in which multiple banks can securely distribute anonymous and untraceable electronic cash. First we give the main ideas, and then we address some of the relevant problems. The basic techniques are due to Chaum [7].

All the banks form a group and their manager will be the country's Central Bank (e.g. U.S. Treasury or the Federal Reserve). If Alice wants to withdraw e-cash from her bank, she first creates an electronic coin C. Her bank applies a *group blind signature* to C and it withdraws the appropriate amount from Alice's account. Alice now gives C and the bank's signature on C to a vendor. The vendor uses the *group public key* to verify the bank's signature on the coin. If the coin is valid, the vendor gives Alice her merchandise, and gives the coin to his bank. The vendor's bank double checks the coin's validity, adds it to a global list of coins that have already been spent (to prevent Alice from double spending), and credits the vendor's account. Now, since Alice's bank is signing blindly, it is possible for Alice to trick her bank by having it sign something other than what it was supposed to sign. We can prevent this by requiring that the bank have different secret signing keys to authorize different dollar amounts. For example, one key could correspond to a \$5 withdrawal, another to a \$10 withdrawal, and so on [15]. There are several other ways to prevent this kind of fraud [21], but we omit them here.

1.5 Advantages of Our Scheme

Previous work on electronic cash systems focused on models in which a *single* bank distributes all the e-cash. We, on the other hand, show how to implement an efficient *multiple* bank model. Our scheme is useful for several reasons. First, the vendor only has to know a single group public key to check the validity of any e-cash he receives. Second, the spender's identity is completely anonymous to both the vendor and his bank since the signature is blind. Third, neither the vendor nor the vendor's bank can determine the user's bank (thus providing an extra layer of anonymity). Fourth, the bank which receives the e-cash just needs to check it with the group public key. Finally, if there are any conflicts regarding which bank issued some piece of e-cash, the group manager (e.g. the U.S. Treasury) can intervene and establish the identity of that bank.

1.6 Disadvantages of Our Scheme

Our scheme is an *online* scheme. That is, the vendor must engage in a protocol with the bank each time he receives an e-coin from Alice. Otherwise Alice could potentially use the same electronic coin for several different transactions because her identity is concealed. This extra interaction between the vendor and bank each time a transaction is made is a disadvantage. It is especially costly for small transactions. We can convert our scheme into an offline scheme if we make a minor modification to our underlying model. We outline this modification later in the paper. Another important drawback of our scheme is that Alice must engage in an expensive several round interactive protocol with her bank each time she wants a new e-coin. Also, our scheme requires $O(l)$ exponentiations per signature, where l is the security parameter. Unfortunately, this parameter must be set to at least 64 for the scheme to be secure. Finally, our scheme does not address issues such as *divisibility* and *transferability*. We leave these issues as open problems.

1.7 Making Our Scheme Offline

We can make our scheme offline under a slight compromise in spender anonymity. In this case we allow for a *passive trustee* who can revoke the anonymity of the spender. To start with, in addition to the banks forming a group, all the spenders in our system form a group – and a trusted third party (passive trustee) can act as the group manager of the spender group. The spender engages in the withdrawal protocol as specified earlier. In addition, after she receives a piece of e-cash, she applies the spender group signature to it. Now, no one, except the group mangager (trusted third party) can determine the identity of the spender, so her identity is still somewhat anonymous. The vendor verifies the validity of e-cash he receives just like he did before, except he must also check that the spender's group signature is authentic. Now, because the identity of the spender is encoded into the cash via a group signature, the vendor doesn't have to worry about dealing with the bank whenever he receives an electronic coin. He can wait until the end of the day and cash all his coins in one shot. If there are any conflicts, then the trusted third party can intervene and determine the identity of the spender. These trusted third parties are commonly refered to as passive trustees because they only have to be present when the user opens her bank account, and when there is a dispute [4].

2 The Group Blind Signature Model

Our group blind signature model extends Camenisch and Stadler's [5] group signature model by adding blindness. Our scheme consists of several signers (members of the group), their group manager, and several users. We combine two concepts: blind signatures [6, 16, 14] and group signatures [5, 9, 3, 10]. Our model allows the members of a group to sign messages on behalf of the group such that the following properties hold for the resulting signature:

1. **Blindness of Signatures:** The signer is unable to view the messages he signs. Moreover, the signer should have no recollection of having signed a particular document even though he can verify that he did indeed sign it. This is new with our scheme.
2. **Unforgeablitiy:** Only group members can issue valid signatures.

3. **Undeniable Signer Identity:** The group manager can always establish the identity of the member who issued a valid signature.
4. **Signer Anonymity:** It is easy to check that a message/signature pair was signed by a group member, but only the group manager can determine which member issued the signature.
5. **Unlinkability:** Two message-signature pairs where the signature was obtained from the same signer cannot be linked.
6. **Security Against Framing Attacks:** Neither the group manager, nor the group members can sign on behalf of other group members.

A group blind signature scheme allows the following five procedures:

Setup: a probabilistic algorithm that generates the group's public key $\mathcal{Y}$ and a secret administration key $\mathcal{S}$ for the group manager.

Join: an interactive protocol between the group manager and the new group member Alice that produces Alice's secret key x, her membership certificate v, and her public key z.

Sign: an interactive protocol between group member Alice and an external user, which, on input message m from the user and Alice's secret key x produces a signature s on m that satisfies the properties above.

Verify: an algorithm which on input $(m, s, \mathcal{Y})$, determines if s is a valid signature for the message m with respect to the group public key $\mathcal{Y}$.

Open: an algorithm which, on input $(s, \mathcal{S})$ returns the identity of the group member who issued the signature s together with a proof of this fact.

Our scheme is the first to extend group signatures to allow for blind signing. As in [5], the following parameters are of interest; all but the last one are constant in the size of the group, but linear in the security parameters:

- the size of the group public key $\mathcal{Y}$.
- the length of signatures.
- the efficiency of the protocols **Setup**, **Join**, **Sign**, **Verify**.
- the efficiency of the protocol **Open**.

Just as in [5] the protocol **Open** can be made to execute in constant time with a compromise in security of the scheme; the basic scheme's **Open** takes time linear in the size of the group.

3 Techniques

In this section, we present some well-studied techniques for proving knowledge of discrete logarithms and related problems. We show how they can be extended to serve as blind signatures of knowledge, which are the building blocks of our group blind signature scheme.

3.1 Preliminaries, Assumptions, Notation

Basic Notation We use "," to denote the concatenation of two (binary) strings, or of binary representations of integers and group elements. We use $c[i]$ to denote the i-th leftmost bit of the string c. For a set A, "$a \in_R A$" means that a is chosen uniformly at random from A. For an integer n, $\mathbb{Z}_n$ denotes the ring of integers modulo n, and $\mathbb{Z}_n^*$ denotes the multiplicative group modulo n.

The Discrete Logarithm Problem and Some Variations Let G be a cyclic group of order n generated by some $g \in G$ (hence $G = \langle g \rangle$). Let $a \in \mathbb{Z}_n^*$. The discrete logarithm of $y \in G$ to the base g is the *smallest* positive integer x such that $g^x = y$. The double discrete logarithm of $y \in G$ to the bases g and a is the smallest positive integer x satisfying:

$$g^{(a^x)} = y \ , \tag{1}$$

if such an x exists. G, g, n, and a can be chosen such that the double discrete logarithm problem is infeasible to solve. An e-th root of the discrete logarithm of $y \in G$ to the base g is an integer x satisfying

$$g^{(x^e)} = y \ , \tag{2}$$

if such an x exists. If the factorization of n is unknown, computing e-th roots in $\mathbb{Z}_n^*$ is assumed to be infeasible [17].

Use of a Hash Function We also assume the existence of an ideal hash function $\mathcal{H}$: $\{0,1\}^* \mapsto \{0,1\}^k$ satisfying the following properties ($\mathcal{H}_l$ denotes the first l bits of $\mathcal{H}$):

1. For a specified parameter l, $\mathcal{H}_l(x)$ contains an equal number of 0's and 1's.
2. $\mathcal{H}_l$ is collision-resistant.
3. $\mathcal{H}$ hides all partial information about its input.

If we have a hash function that satisfies the second and third properties, we can convert it into one which satisfies the first property as well: suppose $\mathcal{H}$ satisfies properties two and three. Consider $\mathcal{H}'(x) = \mathcal{H}(x) \circ \overline{\mathcal{H}(x)}$ where $\overline{\mathcal{H}(x)}$ is the bitwise complement of $\mathcal{H}(x)$ and $\circ$ denotes the concatenation operator. Then $\mathcal{H}'(x)$ has an equal number of 0's and 1's.

We use these assumptions to prove security in the random oracle model [16, 1, 11].

3.2 Blind Signatures of Knowledge

A signature of knowledge is a construct that uniquely corresponds to a given message m that cannot be obtained without the help of a party that knows a secret such as the discrete logarithm of a given $y \in G$ to the base g ($\langle g \rangle =$

G). A proof of knowledge is a way for one person to convince another person that he knows some fact without actually revealing that fact. In a signature of knowledge, the signer ties his knowledge of a secret to the message being signed. A signature of knowledge is used both for the purpose of signing a message and proving knowledge of a secret. Signatures of knowledge were used by Camenisch and Stadler [5]. Their construction is based on the Schnorr signature scheme [19] to prove knowledge. All the signatures of knowledge proposed by [5] can be proved secure in the random oracle model and their interactive versions are zero-knowledge.

To construct blind signatures of knowledge, we modify some of the signatures of knowledge constructs proposed by [5]. The proofs of security that work for [5] also work for our signatures of knowledge.

The first signature of knowledge we consider, both in the regular and blind settings, is the signature of knowledge of the discrete logarithm of a given $y \in G$ to a given base g ($\langle g \rangle = G$).

Definition 1. *An $(l+1)$-tuple $(c, s_1, ..., s_l) \in \{0,1\}^k \times \mathbb{Z}_n^l$ satisfying*

$$c = \mathcal{H}_l(m, y, g, g^{s_1} y^{c[1]}, g^{s_2} y^{c[2]}, \ldots, g^{s_l} y^{c[l]}) \tag{3}$$

is a signature of knowledge of the discrete logarithm of $y \in G$ to the base g on a message m, with respect to security parameter l, denoted

$$SKLOG_l[\alpha \mid y = g^\alpha](m) \tag{4}$$

In general we use Greek letters to represent values whose knowledge will be proven by the signer, and Roman letters to denote values that are known to both the signer and the user.

If the signer does not know the discrete logarithm of y to the base G it is infeasible for him to construct the $l+1$ tuple $(c, s_1, \ldots, s_l)$ satisfying the above equation. We can think of the above definition as an interactive protocol in which the $c[i]$'s represent challenges; the hash function $\mathcal{H}$ serves to remove the interaction. If the prover knows x such that $x = log_g(y)$, he computes $r_1, r_2, ..., r_l \in_R \mathbb{Z}_n$, plugs $m, y, g, g^{r_1}, g^{r_2}, \ldots, g^{r_l}$ to hash function $\mathcal{H}$ to obtain random challenge c, and obtains $s_1, s_2, \ldots, s_l$ by setting

$$s_i = \begin{cases} r_i & \text{if } c[i] = 0 \\ r_i - x(\bmod n) & \text{otherwise} \end{cases} \tag{5}$$

The signature of knowledge constructed above is not blind. There is, however, a way to construct a blind signature that would satisfy Defintion 3.1. Consider the following interactive protocol between a signer and a user:

User Round 0: User wants message m signed and sends a sign request to the signer.
Signer Round 1:

1. Obtain $\{r_i \in_R \mathbb{Z}_n\}$, $1 \le i \le l$.
2. Set $P_i := g^{r_i}$, $1 \le i \le l$.
3. Send $\{P_i\}$ to the user, thus commiting to the $\{P_i$'s$\}$.

User Round 2:

1. Obtain a random permutation $\sigma : \{1, \ldots, l\} \mapsto \{1, \ldots, l\}$ and set $Q_i := P_{\sigma(i)}$, $1 \le i \le l$. (σ will be used to blind the result of $\mathcal{H}$.)
2. Obtain random $a_1, ..., a_l$, and set $R_i := Q_i g^{a_i}$ $1 \le i \le l$. ($\{a_i\}$ are used to blind the inputs to $\mathcal{H}$.)
3. Calculate $c := \mathcal{H}_l(m, y, g, R_1, \ldots, R_l)$.
4. Calculate c' such that $c'[i] = c[\sigma^{-1}(i)]$.
5. Send c' to the signer.

Signer Round 3:

1. Using secret x (recall $x = \log_g y$) compute for

$$1 \le i \le l, t_i = \begin{cases} r_i & \text{if } c'[i] = 0 \\ r_i - x (\bmod\ n) & \text{otherwise} \end{cases} \tag{6}$$

2. Send $\{t_i\}$ to the user.

User Round 4:

1. Verify that $P_i = g^{t_i} y^{c'[i]}$.
2. Compute $s_i := t_{\sigma(i)} + a_i \pmod n$.
3. Output $(c, s_1, ..., s_l)$.

Lemma 1. *The protocol described above produces a blind signature, that is, $(c, s_1, ..., s_l)$ is a signature on m that cannot be linked to the signer's view of the protocol.*

Informal proof:

1. $(c, s_1, ..., s_l)$ is a signature on m, because $c = \mathcal{H}_l(m, y, g, R_1, \ldots, R_i, \ldots, R_l)$, where $R_i = Q_i g^{a_i} = P_{\sigma(i)} g^{a_i} = g^{t_{\sigma(i)}} y^{c'[\sigma(i)]} g^{a_i} = g^{s_i - a_i} g^{a_i} y^{c'[\sigma(i)]} = g^{s_i} y^{c'[\sigma(i)]} = g^{s_i} y^{c[i]}$.
2. $(c, s_1, ..., s_l)$ cannot be linked to the signer's view of the protocol, because σ blinds c and $\{a_i\}$ blind $\{s_i\}$.

Also note that since in the proposed protocol the signer does not provide more information than in the regular proof of knowledge, our protocol is zero-knowledge.

We now define signatures of knowledge based on variations of the discrete logarithm problem and extend the protocol above to construct blind versions of these signatures.

One important variation is the representation problem, studied in [2, 5]. It is a direct generalization of the signature of knowledge discussed above, except we have $y_1, ..., y_w$ instead of just one y, and $g_1, ..., g_v$, instead of just one g. We omit a more formal treatment of signatures of knowledge based on the representation problem due to space limitations.

We also employ signatures of knowledge of the double discrete logarithm and of the roots of discrete logarithms introduced in [5] and show how to make them blind signatures.

Definition 2. *A signature of knowledge of a double discrete logarithm of y to the bases g and a, on message m, with security parameter $l \le k$ denoted*

$SKLOGLOG_l[\alpha \mid y = g^{(a^\alpha)}](m)$, is an $(l+1)$-tuple $(c, s_1, ..., s_l) \in \{0,1\}^l \times \mathbb{Z}^l$ satisfying the equation

$$c = \mathcal{H}_l(m,y,g,a,P_1,\dots,P_l), \;\; where\ P_i = \begin{cases} g^{(a^{s_i})} & if\ c[i]=0 \\ y^{(a^{s_i})} & otherwise \end{cases} \tag{7}$$

It can be shown, in the random oracle model, that a $SKLOGLOG_l[\alpha \mid y = g^{(a^\alpha)}](m)$ can be computed only if a double discrete logarithm $x \in \mathbb{Z}_n$ of the group element $y \in G$ to the bases $g \in G$ and $a \in \mathbb{Z}_n^*$ is known (where $G = \langle g \rangle$ and $|G| = n$). One does not necessarily know the order of $a \in \mathbb{Z}_n^*$, but it is easy to find λ such that $|\mathbb{Z}_n^*| \le 2^\lambda - 1$). Knowing x, λ, and a $\mu > 0$ compute the signature as follows (we use μ to make sure that the distribution of r_i is indistinguishable from the distribution of $r_i - x$, for a secret $x \le 2^\lambda - 1$):

1. For $1 \le i \le l$, generate random $2^\lambda \le r_i \le 2^{\lambda+\mu} - 1$.
2. Set $P_i = g^{(a^{r_i})}$ and compute $c = \mathcal{H}_l(m,y,g,a,P_1,\dots,P_l)$.
3. Set $s_i = \begin{cases} r_i & \text{if } c[i] = 0 \\ r_i - x & \text{otherwise} \end{cases}$

To obtain a blind $SKLOGLOG_l[\alpha \mid y = g^{(a^\alpha)}](m)$, we propose the following protocol, quite similar to the one we use for $SKLOG$:

User Round 0: User wants message m signed and sends a sign request to the signer.
Signer Round 1:
1. For $1 \le i \le l$, generate random $2^\lambda \le r_i \le 2^{\lambda+\mu} - 1$.
2. Set $P_i := g^{(a^{r_i})}$.
3. Send $\{P_i\}$ to the user, commiting to them.

User Round 2:
1. Obtain a random permutation $\sigma : \{1,\dots,l\} \mapsto \{1,\dots,l\}$ and set $Q_i := P_{\sigma(i)}$.
2. For $1 \le i \le l$, generate random $2^{\lambda+\mu} \le b_i \le 2^{\lambda+2\mu} - 1$, and set $R_i := Q_i^{(a^{b_i})}$.
3. Calculate $c := \mathcal{H}_l(m,y,g,R_1,\dots,R_l)$.
4. Calculate c' such that $c'[i] = c[\sigma^{-1}(i)]$.
5. Send c' to the signer.

Signer Round 3:
1. Compute, for $1 \le i \le l$, $t_i = \begin{cases} r_i & \text{if } c'[i] = 0 \\ r_i - x \pmod n & \text{otherwise} \end{cases}$
2. Send $\{t_i\}$ to the user.

User Round 4:
1. Verify that $P_i = \begin{cases} g^{(a^{t_i})} & \text{if } c'[i] = 0 \\ y^{(a^{t_i})} & \text{otherwise} \end{cases}$
2. Compute $s_i := t_{\sigma(i)} + a_i$, $1 \le i \le l$.
3. Output $(c, s_1, ..., s_l)$.

Lemma 2. *The protocol described above produces a blind $SKLOGLOG_l(\alpha \mid y = g^{(a^\alpha)})$.*

Informal proof:

1. $(c, s_1, ..., s_l)$ is a signature on m, because $c = \mathcal{H}_l(m,y,g,a,R_1,\ldots,R_l)$ where

$$R_i = Q_i^{(a^{a_i})} = P_{\sigma(i)}^{(a^{a_i})} = \begin{cases} g^{(a^{t_{\sigma(i)}})(a^{a_i})} = g^{(a^{s_i})} \text{ if } c'[\sigma(i)] = c[i] = 0 \\ y^{(a^{t_{\sigma(i)}})(a^{a_i})} = y^{(a^{s_i})} \text{ if } c'[\sigma(i)] = c[i] = 1 \end{cases}$$

which satisfies the definition of $SKLOGLOG_l$.
2. $(c, s_1, ..., s_l)$ cannot be linked to the signer's view of the protocol, because σ blinds c and $a_1, ..., a_l$ blind $s_1, ..., s_l$.

Definition 3. *A signature of knowledge of an e-th root of the discrete logarithm of y to the base g, on message m, denoted* $SKROOTLOG_l[\alpha \mid y = g^{(\alpha^e)}](m)$ *, is an* $(l+1)$*-tuple* $(c, s_1, \ldots, s_l) \in \{0,1\}^k \times \mathbb{Z}_n^{*\,l}$ *satisfying the following equation:*

$$c = \mathcal{H}_l(m,y,g,e,P_1,\ldots,P_l), \;\; where P_i = \begin{cases} g^{(s_i^e)} & \text{if } c[i] = 0 \\ y^{(s_i^e)} & \text{otherwise} \end{cases}$$

It can be shown that such signature can only be computed if the e-th root of the discrete logarithm x of y to the base g is known; where $x \in \mathbb{Z}_n^*$, $y \in G$, $|G| = n$, $\langle g \rangle = G$, $e \in \mathbb{Z}_n$. When x is known, construct $SKROOTLOG_l[\alpha \mid y = g^{(\alpha^e)}](m)$ as follows:

1. For $1 \le i \le l$, generate random $r_i \in \mathbb{Z}_n^*$.
2. Set $P_i = g^{(r_i^e)}$ and compute $c = \mathcal{H}_l(m,y,g,e,P_1,\ldots,P_l)$.
3. Set $s_i = \begin{cases} r_i & \text{if } c[i] = 0 \\ r_i/x \pmod n & \text{otherwise} \end{cases}$

The $SKROOTLOG_l$ can also be constructed blindly, by applying a similar protocol to the one outlined for $SKLOG_l$ and $SKLOGLOG_l$. We omit this here due to lack of space.

4 Our Group Blind Signature Scheme

Camenisch and Stadler [5] have two schemes for group signatures: their *basic* scheme and their *efficient* scheme. Their *efficient* scheme is less secure and therefore in this extended abstract we concentrate on extending their *basic* scheme to allow for blind signatures. However, our techniques apply to both.

4.1 Setup

To set up the group signature scheme, as in [5], the group manager chooses a security parameter l and computes the following values:

1. An RSA Public Key (n,e), where the length of n is at least $2l$ bits.
2. A cyclic group $G = \langle g \rangle$ of order n for which computing discrete logarithms is hard. In particular, we can choose G to be a cyclic subgroup of $\mathbb{Z}_p^*$ where p is a prime and $n|(p-1)$.
3. An element $a \in \mathbb{Z}_p^*$ where a has large multiplicative order modulo all the prime factors of n.
4. An upper bound λ on the length of the secret keys and a constant $\mu > 1$.

The group's public key is $\mathcal{Y} = (n, e, G, g, a, \lambda, \mu)$.

4.2 Join

As in [5], if Alice wants to join the group she picks a *secret key* $x \in_R \{0, 1, \ldots, 2^\lambda - 1\}$ and calculates $y = a^x \ (mod\ n)$ and the *membership key* $z = g^y$. Alice commits to y and sends (y,z) to the group manager and proves to him that she knows x (without actually revealing x) using techniques similar to the signature of knowledge of discrete logarithm. If the group manager is convinced that Alice knows x he gives her a *membership certificate* $v \equiv (y+1)^{1/e} \ (mod\ n)$. It is an additional security assumption due to [5] that computing v without factoring n is hard.

4.3 Sign and Verify

Unlike [5], our signature construction protocol is blind. When responding to a sign request, the signer does the following:

1. Obtain $q \in_R \mathbb{Z}_n^*$ and set

$$\begin{aligned} \tilde{g} &:= g^q \\ \tilde{z} &:= \tilde{g}^y \end{aligned} \tag{8}$$

2. Obtain random $2^\lambda \leq u_i \leq 2^{\lambda+\mu} - 1$, for $1 \leq i \leq l$ and set

$$P_i^{SKLOGLOG} := \tilde{g}^{(a^{u_i})}, 1 \leq i \leq l \tag{9}$$

3. Obtain random $v_i \in \mathbb{Z}_n^*$, for $1 \leq i \leq l$ and set

$$P_i^{SKROOTLOG} := \tilde{g}^{(v_i^e)}, 1 \leq i \leq l \tag{10}$$

4. Send $(\tilde{g},\ \tilde{z}, \{P_i^{SKLOGLOG}\}, \{P_i^{SKROOTLOG}\})$ to the user.

In turn, the user does the following:

1. Obtains $b \in_R \{1 \ldots 2^\lambda - 1\}$, and $f \in_R \mathbb{Z}_n^*$, and sets $w := (af)^{eb} \pmod n$.
2. Set

$$\begin{aligned} \hat{g} &:= \tilde{g}^w \\ \hat{z} &:= \tilde{z}^w \\ \hat{P}_i^{SKLOGLOG} &:= (P_i^{SKLOGLOG})^w \\ \hat{P}_i^{SKROOTLOG} &:= (P_i^{SKROOTLOG})^w \end{aligned} \tag{11}$$

3. Execute rounds 2, 3, 4 of the blind $SKLOGLOG$ and blind $SKROOTLOG$ protocols, taking $\{\hat{P}_i^{SKLOGLOG}\}$ and $\{\hat{P}_i^{SKROOTLOG}\}$ as commitment values and adjusting the responses $\{t_i^{SKLOGLOG}\}$ and $\{t_i^{SKROOTLOG}\}$ by adding eb for $SKLOGLOG$ and multiplying by $(af)^b$ for $SKROOTLOG$ to obtain

$$\begin{aligned} V_1 &= SKLOGLOG_l[\alpha \mid \hat{z} = \hat{g}^{a^\alpha}](m) \\ V_2 &= SKROOTLOG_l[\beta \mid \hat{z}\hat{g} = \hat{g}^{\beta^e}](m) \end{aligned} \tag{12}$$

The resulting signature on the message m consists of $(\hat{g}, \hat{z}, V_1, V_2)$ and can be verified by checking correctness of V_1 and V_2.

Informal proof of the unforgeability of the signature: It is impossible to construct the signature without the help of a group member, because V_1 proves that the signer must know a membership key, and V_2 proves that the signer also knows the membership certificate that corresponds to that key. That is, V_1 shows that $\hat{z} = \hat{g}^{a^\alpha}$, and therefore:

$$\hat{z}\hat{g} = \hat{g}^{(a^\alpha + 1)} \tag{13}$$

for an integer α that the signer knows. On the other hand, V_2 proves that

$$(a^\alpha + 1) = \beta^e \tag{14}$$

for some β that the signer knows. That can only happen if the signer is a group member, α is his secret key, and β is his membership certificate.

Informal proof of the blindness of the signature: The signer's input into the protocol has been blinded by blinding $\tilde{g}$ and $\tilde{z}$ into random $\hat{g}$ and $\hat{z}$, and constructing two blind signatures of knowledge. Therefore, the resulting signature $(\hat{g}, \hat{z}, V_1, V_2)$ cannot be linked to the signer's view of the protocol.

4.4 Open

Given a signature $(\hat{g}, \hat{z}, V_1, V_2)$ for a message m, the group manager can determine the signer by testing if $\hat{g}^{y_P} = \hat{z}$ for every group member P (where $y_P = \log_g z_P$ and z_P is P's membership key). The group manager can establish the identity of the signer without giving away y_P using the signer's membership key z_P, the signer's commitment to z_P, and a non-interactive proof that $\log_g z = \log_{\hat{g}} \hat{z}$. Since it is considered difficult to test if $\log_{\hat{g}} \hat{z} = \log_{\hat{g}'} \hat{z}'$ members' signatures are anonymous and unlinkable. Thus the running time of this scheme is linear in the size of the group.

4.5 Efficiency of the Proposed Scheme

The proposed scheme is as efficient as the basic group signature scheme proposed by Camenisch and Stadler. That is, for a fixed security parameter l, all of the described operations, except **Open**, take constant time, and the time **Open** takes is linear in the size of the group. All signatures take constant space, and communication complexity per signature is constant. Computational, space, and communication complexities are linear in the security parameter l.

4.6 Security of the Proposed Scheme

Our scheme is exactly as secure as Camenisch and Stadler's Basic Group Signature Scheme [5]. The security of the scheme is based on the assumptions that the discrete logarithm, double discrete logarithm, and the roots of discrete logarithm problems are hard. In addition, it is based on the security of the Schnorr and the RSA signature schemes and on the additional assumption due to [5] that computing membership certificates is hard.

4.7 Improvements on the Efficiency

Camenisch and Stadler [5] also proposed a group signature scheme in which the computational complexity of the **Open** algorithm is constant in the size of the group. This scheme can also be extended to a group blind signature scheme. We chose to omit the details, because the more efficient group signature scheme is also less secure and is known to be broken for certain values of its parameters.

5 Conclusion

We have proposed a group blind digital signature scheme that is secure and efficient, and therefore practical. Our result is an extension of the group signature scheme recently prosposed by Camenisch and Stadler in [5]. We showed how our construction could be used to set up an electronic cash system in which more than one bank can dispense anonymous e-cash.

Acknowledgements

The first author would like to acknowledge the support of DARPA grant DABT63-96-C-0018, the NSF Graduate Research Fellowship and a Lucent Technologies GRPW Program. The second author would like to acknowledge the support of DARPA grant DABT63-96-C-0018. In addition, we thank Ron Rivest for helping us shape our random ideas into a coherent paper. We would never be able to manage without his encouragement. Also, we give many thanks to Eric Lehman, Tal Malkin, Daniele Micciancio, and Amit Sahai for helpful and greatly appreciated discussions. Finally, we thank the anonymous referees for their helpful comments and suggestions.

References

1. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *First ACM Conference on Computer and Communications Security*, pages 62–73, Fairfax, 1993. ACM.
2. Stefan Brands. An efficient off-line electronic cash system based on the representation problem. Technical Report CS-R9323, CWI, April 1993.
3. Jan Camenisch. Efficient and generalized group signatures. In *Proc. EUROCRYPT 97*, pages 465–479. Springer-Verlag, 1997. Lecture Notes in Computer Science No. 1233.
4. Jan Camenisch, Ueli Maurer, and Markus Stadler. Digital payment systems with passive anonymity-revoking trustees. *Journal of Computer Security*, 5(1), 1997.
5. Jan Camenisch and Markus Stadler. Efficient group signatures for large groups. In *Proc. CRYPTO 97*, pages 410–424. Springer-Verlag, 1997. Lecture Notes in Computer Science No. 1294.
6. D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In S. Goldwasser, editor, *Proc. CRYPTO 88*, pages 319–327. Springer-Verlag, 1988. Lecture Notes in Computer Science No. 403.

7. David Chaum. Blind signatures for untraceable payments. In R. L. Rivest, A. Sherman, and D. Chaum, editors, *Proc. CRYPTO 82*, pages 199–203, New York, 1983. Plenum Press.
8. David Chaum. Blind signature system. In D. Chaum, editor, *Proc. CRYPTO 83*, pages 153–153, New York, 1984. Plenum Press.
9. David Chaum and Eugène van Heyst. Group signatures. In *Proc. EUROCRYPT 91*, pages 257–265. Springer-Verlag, 1991. Lecture Notes in Computer Science No. 547.
10. L. Chen and T. P. Pedersen. New group signature schemes (extended abstract). In *Proc. EUROCRYPT 94*, pages 171–181. Springer-Verlag, 1994. Lecture Notes in Computer Science No. 547.
11. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In A.M. Odlyzko, editor, *Proc. CRYPTO 86*, pages 186–194. Springer-Verlag, 1987. Lecture Notes in Computer Science No. 263.
12. S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems. *SIAM. J. Computing*, 18(1):186–208, February 1989.
13. Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Computing*, 17(2):281–308, April 1988.
14. A. Juels, M. Luby, and R. Ostrovsky. Security of blind digital signatures. In *Proc. CRYPTO 97*, Lecture Notes in Computer Science, pages 150–164. Springer-Verlag, 1997. Lecture Notes in Computer Science No. 1294.
15. Laurie Law, Susan Sabett, and Jerry Solinas. How to make a mint: the cryptography of anonymous electronic cash. National Security Agency, Office of Information Security Research and Technology, Cryptology Division, June 1996.
16. David Pointcheval and Jacques Stern. Provably secure blind signature schemes. In M.Y. Rhee and K. Kim, editors, *Advances in Cryptology–ASIACRYPT '96*, pages 252–265. Springer-Verlag, 1996. Lecture Notes in Computer Science No. 1163.
17. Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.
18. B. Schneier. *Applied Cryptography: Protocols, Algorithms, and Source Code in C.* John Wiley & Sons, New York, 1993.
19. C. P. Schnorr. Efficient identification and signatures for smart cards. In G. Brassard, editor, *Proc. CRYPTO 89*, pages 239–252. Springer-Verlag, 1990. Lecture Notes in Computer Science No. 435.
20. Daniel R. Simon. Anonymous communication and anonymous cash. In Neal Koblitz, editor, *Proc. CRYPTO 96*, pages 61–73. Springer-Verlag, 1996. Lecture Notes in Computer Science No. 1109.
21. Peter Wayner. *Digital Cash: Commerce on the Net.* Academic Press, 1996.