Efficient Revocation in Group Signatures

Emmanuel Bresson and Jacques Stern

Ecole Normale Supérieure, 45 rue d'Ulm, 75230, Paris, France E-mail:
{Emmanuel.Bresson,Jacques.Stern}@ens.fr.

Abstract. We consider the problem of revocation of identity in group signatures. Group signatures are a very useful primitive in cryptography, allowing a member of a group to sign messages anonymously on behalf of the group. Such signatures must be anonymous and unlinkable, but a group authority must be able to open them in case of dispute. Many constructions have been proposed, some of them are quite efficient. However, a recurrent problem remains concerning revocation of group members. When misusing anonymity, a cheating member must be revoked by the authority, making him unable to sign in the future, but without sacrifying the security of past group signatures. No satisfactory solution has been given to completely solve this problem. In this paper, we provide the first solution to achieve such action for the Camenish-Stadler [6] scheme. Our solution is efficient provided the number of revoked members remains small.

1 Introduction

1.1 Overview of group signatures

Digital signatures are becoming a fact of life. They are used in more and more products and protocols and one can find a large amount of literature dealing with their applications, variants and security [12, 1, 2]. Group signatures were first introduced in 1991 by Chaum and Van Heyst [8]. This recent concept is linked (at least originally) with applications to electronic cash. It tries to combine security (no framing, no cheating) and privacy (anonymity, unlinkability). These two constraints have recently motivated much work and many publications, to make such protocols more realistic and efficient.

A digital group signature scheme deals with a group, possibly a dynamic one, whose users are called *players* (or simply *members*) and most of the time a *group center* (also called *group leader*), who is the authority with ability to "open" a signature in case of later dispute, and to reveal the identity of the actual signer. The underlying group structure is said to be *dynamic* if the number of users can increase by registering and adding new members.

1.2 Previous work

The concept of group signatures was introduced in 1991 in [8]. That paper proposed four different group signature schemes. The first one provided unconditional anonymity, and the others provided computational anonymity only. However, adding new members was not always possible, and in some schemes, the leader needed to contact group members in order to open a group signature. See [8] and [13] for a comparison of these schemes.

At Eurocrypt'94, Chen and Pedersen proposed two new schemes, based on undeniable signatures [9]. They used proofs of knowledge of discrete logarithm to build group signatures: proving the knowledge of a discrete logarithm within a collection, without revealing which one is known, corresponds to the requirements of a group signature: proving membership without revealing individual's identity.

Unfortunately, all the above schemes were relatively inefficient due to a growth of the signature size linear with respect to the number of group members. A solution has been proposed by Camenish and Stadler in 1997 [6]. Their scheme provides a constant-size signature a well as a constant-size group public key. The tools they used to build such a scheme are an ordinary digital signature scheme, a probabilistic semantically secure encryption scheme and a one-way function. We recall the description and the functioning on this scheme in section 3.

1.3 Functioning and Security

We now give a more formal definition of a group signature scheme, as well as the related security requirements.

A group signature scheme allows members to sign on behalf of the group. That is, any user (not necessarily a member) should be able to verify that the message has been signed by an authorized member of the group (i.e. a registered member). However, the verifier should learn no information on which member actually signed the message. Moreover, the signatures must be unlinkable, that is, deciding whether two different signatures have been produced by the same person must be (computationaly) infeasible. In case of dispute, the verifier can interact with the group leader to get the real identity of the actual signer.

More formally, a group signature scheme consists of the following algorithms:

- `SETUP`: a probabilistic algorithm initializing public parameters and providing a secret key to the group leader.
- `JOIN`: an interactive protocol between the group center and a user becoming a group member. This protocol provides a secret key and a membership key to the new member, and registers his identity.
- `SIGN`: a probabilistic algorithm, computing from a message m and a member's secrets s a group signature σ.
- `VERIF`: an algorithm, run by any user, which checks that a signature σ has been produced by an authorized signer.
- `OPEN`: an algorithm allowing the group leader to obtain the identity of the member who actually signed a given message.

These algorithms are considered in the case of dynamic group. Otherwise, there is no `JOIN` algorithm, and each member receives his keys in the `SETUP` algorithm. Note that there exist many variants of group signatures (see [14, 13, 15, 5, 3]). Depending on what additional properties are proposed, we could find corresponding variants in the algorithms.

The following conditions must hold for a group signature scheme:

- **Correctness:** Any signature generated by a registered group member is valid.
- **Unforgeability:** Only registered members are able to sign messages.
- **Anonymity** (or untraceability)**:** Identifying the signer of a given signature is computationally hard, except for the group manager.
- **Unlinkability:** Deciding whether two signatures were generated by the same member is computationally hard.
- **Traceability:** Any fairly-generated signature can be opened by the group leader in order to identify the actual signer.
- **Exculpability** (or unforgeability of tracing, or no framing)**:** No coalition of members nor the group leader can sign on behalf of other members, which means that they cannot compute a signature that can be associated to another group member.
- **Coalition-resistance** (or unavoidable traceability)**:** No coalition of members can prevent a group signature from being opened. A scheme offering provable security against coalition resistance was proposed in [1]

One critical point in group signatures is the efficiency of the algorithms. In particular, one wants to avoid the group public key or the signature size to be linear in the number of group members. This is especially true in very large group as well as very dynamic ones. Efficiency of `SIGN` and `VERIF` algorithms is also important.

1.4 Motivation of our work

Revocation in group signatures is a very delicate problem. In a paper by Ateniese and Tsudik [3], some critical points are put forth: coalition-resistance and member deletion. In this paper, we concentrate on the second one.

In some cases, it can be useful to *delete* members from a group. This can be necessary for many reasons (cheating from the said member, e.g.), and one does not want to change the group public key as well. Revocation of a member should prevent him from generating valid group signatures in the future. At the same time, one generally wants to preserve his past signatures, that is, keeping them indistinguishable from others signatures, unlinkable, openable, etc. The difficulty encountered can be stated as follows. On the one hand, in order to preserve anonymity, group signatures must not need to be opened when checking the legitimity of the signer: verifying that the actual signer is not a revoked member must be feasible by anybody, in a public manner and without the help of the group leader. The verifier must learn nothing about the signer but the fact he is not a deleted member. On the other hand, in order to preserve anonymity and unlinkability of **past** signatures, we require that no private information (who could help somebody to link signatures) concerning revoked members be published. Of course, the guarantee of opening signatures in case of conflict remains.

Our paper is organized as follows. In section 2, we describe the basic tools used in the Camenish/Stadler scheme, this later being exposed in section 3.

In section 4, we explain our technique to achieve revocation of members in that scheme. We propose a solution efficient if the number of deleted member is small, the size of the group signature growing linearly with that number. Finally, we discuss the security of the scheme and conclude.

2 Signatures of knowledge

Many group signature schemes use the notion of *signature of knowledge*. This cryptographic tool allows one party to prove the knowledge of a secret value, without revealing any information on it. Such tools are zero-knowledge proofs of knowledge and minimum-disclosure proofs. The notion of signature of knowledge is based (originally) on the Schnorr digital signature scheme [16]. We call them signature of knowledge instead of proofs of knowledge to avoid confusion with zero-knowledge proofs while reminding the fact they are based on signature schemes (being message-dependent). Let us review the most important signatures of knowledge one can find in the area of group signatures.

In the following sections, we will denote by Greek letters the values whose knowledge is proven and by Latin or any other symbol the elements that are publicly known. We consider a cyclic group G, of order n (where n is an RSA modulus) and a random element g generating G. We consider also a hash function $\mathcal{H}$ from $\{0,1\}^*$ to $\{0,1\}^k$ (k being typically equal to 160). All security notions are considered in the Random Oracle model [4].

2.1 Knowledge of a discrete logarithm

Given an element $y \in G$, a signature of knowledge of the discrete logarithm of y to the base g on the message m is a pair $(c, s) \in \{0,1\}^k \times \mathbb{Z}_n^*$ satisfying:

$$c = \mathcal{H}(m \| y \| g \| g^s y^c)$$

This signature is denoted by:

$$SKLOG[\alpha : y = g^\alpha](m)$$

Suh a pair can be computed by a prover who knows the secret value x (such that $y = g^x$ holds) as follows: first choose a random value $r \in \mathbb{Z}_n$ and compute c as $c := \mathcal{H}(m \| y \| g \| g^r)$. Knowing x, it is possible to compute $s := r - xc$.

2.2 Knowledge of a representation

Consider another element $h \in G$ whose discrete logarithm to the base g is unknown. Given an element $y \in G$, a signature of knowledge of a representation of y to the bases g and h on the message m is a tuple $(c, s_1, s_2) \in \{0,1\}^k \times \mathbb{Z}_n^{*2}$ satisfying:

$$c = \mathcal{H}(m \| y \| g \| h \| g^{s_1} h^{s_2} y^c)$$

Such a tuple is denoted by:

$$SKREP[\alpha, \beta : y = g^{\alpha} h^{\beta}](m)$$

A prover who knows a representation (x_1, x_2) of y to the bases g and h can compute an accepting tuple as follows: at first, choose two random numbers $r_i \in \mathbb{Z}_n, \quad i = 1, 2$ and compute $c = \mathcal{H}(m \| y \| g \| h \| g^{r_1} h^{r_2})$. Then, the values s_i can be constructed as $s_i = r_i - x_i c, \quad i = 1, 2$. This construction can easily be extended to more than one element and two bases [6].

2.3 Knowledge of roots of representation

Such signatures are used to prove that one knows the e-th root of a part of a representation. That is, given an element $y \in G$, one wants to prove knowledge of a pair (α, β) such that the equation $y = h^{\alpha} g^{\beta^e}$ holds. Such proofs have been proposed by Camenish and Stadler in [6]. They can be used to improve efficiency of signature of knowledge for double discrete logarithm and roots of discrete logarithms, these proofs being bit-to-bit process and then quite inefficient. See [6] for further details.

Given an element $y \in G$ and an small integer e, a signature of knowledge of the e-th root of the g-part of the representation of y to the bases g and h on the message m, consists in an $(e-1)$-tuple $(y_1, \ldots, y_{e-1}) \in G^{e-1}$ and a signature of knowledge of representation of $(y_1, \ldots, y_{e-1}, y)$ to the bases $\{h, g\}, \{h, y_1\}, \ldots, \{h, y_{e-1}\}$ respectively. More precisely, the latter signature of knowledge is:

$$\begin{aligned} SKREP[\gamma_1, \ldots, \gamma_e, \delta : y_1 = h^{\gamma_1} g^{\delta} \quad \wedge \quad y_2 = h^{\gamma_2} y_1^{\delta} \quad \wedge \quad \ldots \\ \wedge \quad y_{e-1} = h^{\gamma_{e-1}} y_{e-2}^{\delta} \quad \wedge \quad y = h^{\gamma_e} y_{e-1}^{\delta}](m) \end{aligned}$$

where $\wedge$ is a conjunction's symbol. This means that all relations specified within square brakets [..] are proven. Note that there exist proofs of knowledge for disjunctive relations or more complicated statements.

Knowing secret values a and b such that $y = h^a g^{b^e}$, one can efficiently compute the desired signature. With randomly chosen numbers r_i, for $i = 1, \ldots, e$–1, first calculate the $(e-1)$-tuple: $y_i := h^{r_i} g^{b^i}$. According to the above equations, by identifying the representations of each y_i to the bases h and y_{i-1}, we actually have: $\gamma_1 = r_1$, $\gamma_i = r_i - br_{i-1}$ for $i = 2, \ldots, e-1$, $\gamma_e = a - br_{e-1}$ and $\delta = b$. The sub-signature of representation is as follows: c is computed as

$$c = \mathcal{H}(m \| g \| h \| y_1 \| \cdots \| y_{e-1} \| h^{t_1} g^d \| h^{t_2} y_1^d \| \cdots \| h^{t_e} v_{e-1}^d)$$

where $t_1, \ldots, t_e, d$ are random numbers in $\mathbb{Z}_n$. Then "answers" are computed as usual:

$$\begin{aligned} s_1 &= t_1 - c\gamma_1 \\ s_2 &= t_2 - c\gamma_2 \\ &\ldots \end{aligned}$$

$$\begin{aligned} s_e &= t_e - c\gamma_e \\ s_d &= d - c\delta \end{aligned}$$

This signature of knowledge of a representation of $(y_1, y_2, \ldots, y_{e-1}, y)$ to the respective bases $\{h, g\}$,$\{h, y_1\}, \ldots, \{h, y_{e-1}\}$ consists in the tuple $(c, s_1, \ldots, s_e, s_d)$ and is checked by verifying the following equation:

$$c = \mathcal{H}(m\|g\|h\|y_1\|\cdots\|y_{e-1}\|y_1^c h^{s_1} g^{s-d}\|y_2^c h^{s_2} y_1^{s_d}\|\cdots\|y^c h^{s_e} y_{e-1}^{s_d})$$

The global signature is denoted by:

$$SKROOTREP[\alpha, \beta : y = h^{\alpha} g^{\beta^e}](m)$$

The following equations show what is checked by the verifier:

$$\begin{aligned} y_1 &= h^{\gamma_1} g^{\delta} \\ y_2 &= h^{\gamma_2} y_1^{\delta} &&= h^{\gamma_2 + \gamma_1 \delta} g^{\delta^2} \\ y_3 &= h^{\gamma_3} y_2^{\delta} &&= h^{\gamma_3 + \gamma_2 \delta + \gamma_1 \delta^2} g^{\delta^3} \\ \cdots &= \cdots \\ y &= h^{\gamma_e} y_{e-1}^{\delta} &&= h^{\gamma_e + \cdots + \gamma_2 \delta^{e-2} + \gamma_1 \delta^{e-1}} g^{\delta^e} \end{aligned}$$

Hence, y is actually of the form $h^{\alpha} g^{\beta^e}$, where α and β are proven to be known by the signer.

2.4 Knowledge of roots of discrete logarithms

We can use the previous tool to construct efficient signature of knowledge of roots of discrete logarithm. Given an element $y \in G$, an small integer e and two generators g and h of G (such that the discrete logarithm of h to the base g is unknown), a signature of knowledge of the e-th root of the discrete logarithm of y to the base g on the message m consists of two signatures:

$$SKREP[\delta : y = g^{\delta}](m) \quad \text{and} \quad SKROOTREP[\alpha, \beta : y = h^{\alpha} g^{\beta^e}](m)$$

Such a proof is checked by verifying the correctness of the two underlying signatures. Since the prover can know at most one representation of y to the bases g and h (otherwise, he would be able to compute $\log_g h$), it follows that: $\alpha \equiv 0 \pmod{n}$ and $\delta \equiv \beta^e \pmod{n}$. Hence the verifier must be convinced that the prover knows a e-th root of the discrete logarithm of y to the base g.

Such a signature is denoted:

$$SKROOTLOG[\eta : y = g^{\eta^e}](m)$$

3 Group signatures by Camenish and Stadler

3.1 System overview

The system parameters are chosen as follows by the group manager during the setup procedure:

- n is an RSA modulus; e_1 and e_2 are two public RSA exponents (and thus relatively prime to $\varphi(n)$).
- $G = \langle g \rangle$ is a cyclic group generated by g of order n.
- $h \in G$ is an element whose discrete logarithm to the base g is unknown.
- f_1 and f_2 are two elements in $\mathbb{Z}_n \backslash \{0, 1\}$.
- $R = h^w$, for a randomly chosen $w \in \mathbb{Z}_n$ is the manager's public key.

The group leader should keep secret the factorization of n as well as the value of w. All others parameters are public and consitute the group's public key.

Security hypothesis. System parameters should be chosen in such a way that the following conditions hold (see in 5 the proof of security).

- Computing discrete logarithm to the base g should be infeasible in G. This can be achieved by choosing for G a subgroup of $\mathbb{Z}_p^*$, where p is a prime number and $n|(p-1)$.
- The discrete logarithm of h to the base g is unknown (and hard to compute).
- Both e_1-th and e_2-th roots of f_1 as well as those of f_2 are unknown (and hard to compute without the factorization of n).

3.2 Member registration

Consider a user Alice who wants to become a member of the group. She first has to compute her *membership key*: she chooses a random number $x \in \mathbb{Z}_n^*$. Let $y = x^{e_1} \pmod{n}$. Alice keeps y and x secret as the private parts of her membership key. Then Alice computes $z = g^y$ and publishes it together with her identity. This is the public part of her membership key.

Alice must then register these values to the group manager in order to get a *membership certificate*. She cannot send y to the group manager, otherwise he could forge Alice's signatures as he wants. Thus she sends z, a blinded value of y and a proof that this value actually blinds a well-formed membership key. In order to do that, Alice computes:

$$\begin{aligned}
&\tilde{y} := r^{e_2}(f_1 y + f_2) \pmod{n} \quad \text{for} \quad r \in_R \mathbb{Z}_n^* \\
&U := SKROOTLOG[\alpha : z = g^{\alpha^{e_1}}](\text{` '}) \\
&V := SKROOTLOG[\beta : g^{\tilde{y}} = (z^{f_1} g^{f_2})^{\beta^{e_2}}](\text{` '})
\end{aligned}$$

and sends $z, \tilde{y}, U, V$ to the manager. If both U and V are correct, the latter should be convinced that $\tilde{y}$ actually blinds a correct membership key, contained in the value z (α and β proving indeed the knowledge of x and r respectively). Then the manager computes a blinded version of the membership certificate as:

$$\tilde{v} := \tilde{y}^{1/e_2} \pmod{n}$$

The (unblinded) membership certificate is

$$v = \tilde{v}/r = (f_1 y + f_2)^{1/e_2}$$

A possible choice for parameters is suggested in [6]: $e_1 = 5, e_2 = 3, f_1 = 1, f_2$ is such that 3rd root is hard to compute. It seems to be difficult to find some tuples (x, v) such that $v^{e_2} = f_1 x^{e_1} + f_2$ holds, without knowing the factorisation of n. This assumption is used in the proof of security of our scheme 5.1.

3.3 Signing messages

To sign a message m, Alice basically computes, dependent on m, signatures of knowledge proving that she is a registered member (this allows the signature to be verified). At the same time, she encrypts her membership key z with respect to the group manager's public key (this allows the signature to be opened). To this aim, Alice chooses a random number $r \in \mathbb{Z}_n^*$ and sends the following five elements as the signature of m:

$$\begin{aligned}
\tilde{z} &:= h^r g^y \\
d &:= R^r \\
V_1 &:= SKROOTREP[\alpha, \beta : \tilde{z} = h^\alpha g^{\beta^{e_1}}](m) \\
V_2 &:= SKROOTREP[\gamma, \delta : \tilde{z}^{f_1} g^{f_2} = h^\gamma g^{\delta^{e_2}}](m) \\
V_3 &:= SKREP[\epsilon, \zeta : d = R^\epsilon \quad \wedge \quad \tilde{z} = h^\epsilon g^\zeta](m)
\end{aligned}$$

The correctness of the group signature is the conjunction of the correctness of V_1, V_2 and V_3. Indeed, considering V_1 together with V_2, and assuming that Alice can know at most one representation of $\tilde{z}^{f_1} g^{f_2}$ to the bases g and h, the verifier is convinced that:

$$\gamma = \alpha f_1 \pmod{n} \qquad \text{and} \qquad \delta^{e_2} = f_1 \beta^{e_1} + f_2 \pmod{n}$$

The second equation proves that Alice knows a valid membership certificate $v = \delta$ whose related secret membership key is $x = \beta$. Now considering V_3, it proves that the same random number is used in the computation of $\tilde{z}$ and d. Therefore $(d, \tilde{z})$ is an El-Gamal encryption of $z = g^y$ with respect to the leader's public key (h, R) (the secret key being actually $1/w$ rather than w). If V_3 is correct, the encryption is well-formed, ensuring that the signature can be opened if necessary.

3.4 Opening signatures

As just said, the opening of signature consists of the decryption of $(d, \tilde{z})$ as an ElGamal ciphertext. By computing $\hat{z} = \tilde{z}/d^{1/w}$, the group center obtains the public membership key z of the actual signer. To prove such a fact, he can produce a signature of knowledge of the representation of $\tilde{z}, h$ to the bases $\{z, d\}, \{R\}$ respectively, that is:

$$SKREP[\omega : \tilde{z} = zd^{\omega} \quad \wedge \quad h = R^{\omega}](`\ ')$$

where ω holds for $1/w$.

4 Achieving revocation of identity

4.1 Introduction

Revocation of identities (or members deletion) is a very delicate problem. Ateniese and Tsudik [3] have suggested that Certificate Revocation Lists (CRLs) is not an appropriate method for group structures. They invoked the following reasons: firstly, since group signatures are based on anonymous and unlinkable mechanisms, the fact that a given signature was made (illegally) by a revoked member can be only proven by the group manager, by opening the signature. This is surely not practical. Secondly, if the group center reveals some informations or secret values concerning a revoked member, in order to immediately detect possible further cheating, how can the anonymity and unlinkability of his past signatures be preserved? Thirdly, decision of changing the group's public-key is clearly not desirable in very large groups, or in groups with frequent membership changes.

4.2 Our approach

In this section, we propose a solution to delete members from a group without leaking any information about their past signatures. In case of member deletion, the group manager would issue a list of identities (public membership keys "z") and would certify them as being deleted (for instance by signing the list). Any user could continue to sign if he is able to prove, in a zero-knowledge way, that his membership key contained in the signature is not present in the revocation list. It is clear that, while releasing only public informations, the process leaks no extra information and thus does not compromise the past signatures of the deleted members. The drawback is that signature size will grow linearly with respect to the number of members deleted. Providing a constant-size revocation mechanism remains an open and interesting challenge.

4.3 Proving a non-encryption of a given value

We show here how to prove that the encrypted value in an ElGamal ciphertext is not equal to a particular one. More precisely, we can prove that the discrete logarithm of the plaintext is known and that the plaintext differs form a particular value. Consider the ElGamal cryptosystem in a group $H = \langle h \rangle$ of order a large prime number p, and let $y = h^x \pmod{p}$ be the public key associated with the secret key x. A message m is encrypted by $(A, B) = (h^r, y^r m)$, where r is a random number. Let $\bar{m}$ be a particular message. We now explain how the sender can publicly prove that the encrypted message m is different from a value $\bar{m}$, in the case where $m = g^u \pmod{p}$.

We propose a technique using a "witness" value. The idea is quite similar to that used by Canetti and Goldwasser. In [7], they propose a method to distribute the Cramer-Shoup cryptosystem [10]. See [7] for more details. In the context of group members revocation, we first note that the problem can be stated as follows: the signer publishes a random power of $m/\bar{m}$ as a witness together with a proof that this witness is well-constructed and that the plaintext equals the numerator of that underlying fraction, that is m. The fact that the witness value differs from 1 thus proves that the plaintext differs from $\bar{m}$. More formally, the sender computes the following values, where r and r' are random:

$$\begin{aligned} (A, B) &= (h^r, y^r m) && : \text{the ciphertext} \\ t &= (m/\bar{m})^{r'} && : \text{the witness} \\ V &= SKREP[\alpha, \beta, \gamma, \delta : A = h^\alpha \quad \wedge \quad B = y^\alpha g^\beta \\ &\qquad\qquad \wedge \quad A^\gamma = h^{-\delta} \quad \wedge \quad t = (B/\bar{m})^\gamma y^\delta](`') \end{aligned}$$

What does this proof show? The first two equations simply prove that the same value α is used to compute A and B, and thus that (A, B) is an encryption of $m = g^\beta$ with respect to the public key $y = h^x$. This guarantees the ciphertext is fairly computed and that the discrete logarithm of the plaintext is known. Now considering the first and third equations in the proof:

$$A = h^\alpha \quad \text{and} \quad A^\gamma = h^{-\delta},$$

we obtain, taking the discrete logarithm of A^γ to the base h:

$$\delta = -\alpha\gamma \pmod{n}$$

Replacing this value in the last equation, we get:

$$t = \left(\frac{B}{\bar{m}}\right)^\gamma y^{-\alpha\gamma} = \frac{(By^{-\alpha})^\gamma}{\bar{m}^\gamma} = \left(\frac{m}{\bar{m}}\right)^\gamma$$

Being convinced of this equality, the fact that $t \neq 1$ proves that $m \neq \bar{m}$.

4.4 Application to a revocation mechanism

In this paragraph, we use the previous technique to construct a revocation mechanism in the group signature scheme by Camenish and Stadler [6]. We first consider the basic case, where only one member has been revoked.

Recall how the mechanism to open group signatures works. The signer (Alice) encrypts her identity (z) according to the ElGamal scheme and with respect to the group manager public key (h, R). Thus, the manager is able to reveal her identity by decrypting this ciphertext. The signature of knowledge V_3 is used to publicly ensure that the encryption is well-formed: the ciphertext is $(d, \tilde{z})$ where $d = R^r$, $\tilde{z} = zh^r$; V_3 convinces any verifier that the same random number r is used in d and $\tilde{z}$.

Using the fact that the plaintext is Alice's identity, and thus can be written in the desired form g^{y_A}, we can apply our technique to slightly modify the proof V_3 in order to convince the verifier of the group signature that the identity of the signer, say z, differs from a publicly revoked value z_1. We also add the "witness" value t (we will have to transmit several witnesses in case of multiple revocations); other items in the group signature remain unchanged.

$$\begin{aligned}
\tilde{z} &:= h^r g^y \\
d &:= R^r \\
t &:= (z/z_1)^{r'} \text{ for some random number } r' \\
V_1 &:= SKROOTREP[\alpha, \beta : \tilde{z} = h^\alpha g^{\beta^{e_1}}](m) \\
V_2 &:= SKROOTREP[\gamma, \delta : \tilde{z}^{f_1} g^{f_2} = h^\gamma g^{\delta^{e_2}}](m) \\
V_3 &:= SKREP[\epsilon, \zeta, \eta, \lambda : d = R^\epsilon \quad \wedge \quad \tilde{z} = h^\epsilon g^\zeta \\
&\qquad \wedge \quad d^\eta = R^{-\lambda} \quad \wedge \quad t = (\tilde{z}/z_1)^\eta h^\lambda](m)
\end{aligned}$$

If the three proofs V_1, V_2, V_3 are correct, the verifier is convinced, as in the classical scheme, that the encryption of z is well-formed, that is $(d, \tilde{z})$ is an ElGamal encryption of z. According to V_3, the verifier can deduce as explained above:

$$\lambda = -\eta\epsilon \pmod{n}$$

And then, by replacing these value in the last equation of V_3, he obtains:

$$t = \left(\frac{\tilde{z}}{z_1}\right)^\eta h^{-\eta\epsilon} = \frac{(\tilde{z}h^{-\epsilon})^\eta}{z_1^\eta} = \left(\frac{z}{z_1}\right)^\eta$$

The verifier is convinced of the existence of a value η such that the above equation holds. Granted this, the fact that $t \neq 1$ actually proves that $z \neq z_1$. Hence, Alice is not the revoked member.

4.5 Case of multiple revocations

We can easily extend this feature to the scenario of multi-revocations. However, as observed above, the size of the signature will grow linearly with the number of members deleted. More precisely, the number of values t having to been transmitted will be proportional (and even equal) to the number of members revocated. On the other hand, the size of the signature of knowledge V_3 will not grow any more.

Let us consider a list $\mathcal{L}$ of l deleted members, whose identities (or public membership keys) are denoted $z_1, \ldots, z_l$. If a signer Alice wants to sign a message m while proving she is not in the list of revocated members, she will send together with $\tilde{z}$ and d the following l values:

$$t_1 = (z/z_1)^{r'}, \ldots, t_l = (z/z_l)^{r'}$$

where r' is a random number. The proofs V_1 and V_2 remain unchanged, while V_3 becomes:

$$SKREP[\epsilon, \zeta, \eta, \lambda : d = R^{\epsilon} \quad \wedge \quad \tilde{z} = h^{\epsilon} g^{\zeta} \quad \wedge \quad d^{\eta} = R^{-\lambda}$$
$$\wedge \quad t_1 = (\tilde{z}/z_1)^{\eta} h^{\lambda} \quad \wedge \ \ldots \ \wedge \quad t_l = (\tilde{z}/z_l)^{\eta} h^{\lambda}](m)$$

It is important to note that the number of "equations" in V_3 does not change the length of V_3 itself. V_3 is made of a tuple (c, s_1, s_2, s_3, s_4) corresponding to a "challenge" and four "answers" since one wants to prove the knowledge of four private values. The only data which grows when increasing the revocation list are the transmitted "witnesses" $t_1, \ldots, t_l$.

It is also important to notice that the constant size of V_3 is due to that we use the same random r' in all the witness values. We claim that this can be done without loss of security. Consider the case $l = 2$; denote $\mathcal{S} = (z_1, z_2, t_1, t_2)$, where $t_1 = (z/z_1)^r, t_2 = (z/z_2)^r$, the distribution which appears to the verifier in the scheme. It is esay to show the distribution $\mathcal{S}$ is as indistinguishable from a random distribution as the Diffie-Hellman distribution $\mathcal{D} = (g, g^a, g^r, g^{ar})$. To do so, let

$$g = \frac{z}{z_1} \quad \text{and} \quad a = \log_g \left(\frac{z}{z_2}\right)$$

Then we have: $z/z_2 = g^a$ and we can rewrite:

$$\mathcal{S} = (z_1, z_2, t_1, t_2) = (zg^{-1}, zg^{-a}, g^r, g^{ar}) \overset{c}{\approx} (g, g^a, g^r, g^{ar}) = \mathcal{D}$$

where $\overset{c}{\approx}$ stands for "computationally indistinguishable".

5 Security of the enhanced scheme

5.1 Correctness and unforgeability

Verifying correctness is trivial. Since the validity of a group signature is checked by verifying the three proofs of knowledge V_1, V_2, V_3, it is obvious that a registered member of the group is able to produce valid sinatures (keep in mind that the quantities $\alpha, \beta, \gamma, \delta, \epsilon, \zeta$ represent r, x, rf_1, v, r, y respectively, as defined in section 3.3).

Unforgeability against adaptive chosen-message attacks

We now prove that unforgeability is satisfied against an active adversary. We consider a polynomial-time bounded adversary having access to a signing oracle. A signing oracle for group signatures can be modelled as follow: the adversary makes a query to the oracle and obtains a group signature on a message of his choice. The signing oracle returns a valid group signature, which means that this later can be opened by the manager. We show that the identity revealed by such hypothetical opening does not influence our proof.

In that model, the adversary makes a polynomial number of queries to obtain adaptively some group signatures on messages of his choice. Next, the adversary tries to produce a valid group signature. We say that he is successful if he can output a message m^* and a valid group signature $(\tilde{z}^*, d^*, V_1^*, V_2^*, V_3^*)$ and if m^* was not previously queried to the signing oracle. The security of the group signature scheme states that this occurs with negligible probability.

It can be shown using standard techniques that, in the Random Oracle model, we can efficiently simulate the signing oracle used in a chosen-message attack. For instance, the signature of knowledge denoted by V_3 :

$$SKREP[\epsilon, \zeta, \eta, \lambda : d = R^{\epsilon} \ \wedge \ \tilde{z} = h^{\epsilon} g^{\zeta} \ \wedge \ d^{\eta} = R^{-\lambda} \ \wedge \ t = (\tilde{z}/z_1)^{\eta} h^{\lambda}](m)$$

is a tuple (c, s_1, s_2, s_3, s_4) satisfying:

$$c = \mathcal{H}(m \| d \| \tilde{z} \| R \| h \| g \| d^c R^{s_1} \| \tilde{z}^c h^{s_1} g^{s_2} \| d^{s_3} R^{s_4} \| t^c (\tilde{z}/z_1)^{s_3} h^{s_4})$$

Such a tuple can be simulated as follows (notice than we need the value of t to correctly simulate V_3):

```
SIMULATE-SKREP
1  Choose $s_1, s_2, s_3, s_4, c$ at random
2  Define $\mathcal{H}(m \| d \| \tilde{z} \| R \| h \| g \| d^c R^{s_1} \| \tilde{z}^c h^{s_1} g^{s_2} \| d^{s_3} R^{s_4} \| t^c (\tilde{z}/z_1)^{s_3} h^{s_4}) := c$
3  **Return** $c, s_1, s_2, s_3, s_4$ as the signature of knowledge
```

Now we show the security of the scheme. Assume that, at the end of the previously described game, the adversary outputs a valid group signature

$$(\tilde{z}^*, d^*, V_1^*, V_2^*, V_3^*)$$

for which the verification algorithm outputs "`Valid`".

The correctness of V_1^* and V_2^* ensures that he knows four values α, β, γ and ζ such that the following equations hold:

$$\tilde{z}^* = h^{\alpha} g^{\beta^{e_1}} \quad , \quad \tilde{z}^{*f_1} g^{f_2} = h^{\gamma} g^{\delta^{e_2}}$$

which implies:

$$\tilde{z}^{*f_1} g^{f_2} = h^{\gamma} g^{\delta^{e_2}} = h^{\alpha f_1} g^{f_1 \beta^{e_1} + f_2}$$

Hence, we have two representations of $\tilde{z}^{*f_1} g^{f_2}$ to the bases g and h. Consequently, either the two representations are different and the adversary can compute $\log_g h$, or they are identical and we have $\gamma = \alpha f_1 \quad , \quad \delta^{e_2} = f_1 \beta^{e_1} + f_2$, which means that he had computed a certificate, δ, without registering the corresponding key β. Both of these scenarios are assumed to occur with negligible probability. This concludes our proof.

5.2 Anonymity and unlinkability

Anonymity is ensured by the security of the ElGamal scheme, that is, the hardness of computational Diffie-Hellman problem. It is easy to see that, because, since V_1, V_2, V_3 are zero-knowledge, the only information an adversary has to learn z is the encryption $(\tilde{z}, d)$ of it.

More interesting is unlinkability. We can prove that the signatures are unlinkable by using a signature distinguisher as an oracle to break the decisional Diffie-Hellman problem, or, which is equivalent, the semantic security of ElGamal scheme.

Assume we have an oracle that can distinguish two group signatures, i.e. that can win with non-negligible probability the following game: a message m and two members z_1 and z_2 are chosen. A bit b is secretly and randomly chosen. Then the group member z_b signs the message m. The resulting signature $(\tilde{z}, d, V_1, V_2, V_3)$ is given to the adversary which outputs a bit b'. He wins if $b = b'$.

We now can use such an adversary to break the semantic security of ElGamal [11]. Consider the following two algorithms:

```
FINDER
Randomly choose $z_1, z_2$ in $G$
```

```
DISTINGUISHER($A, B$)
/*$(A, B) = (h^r z_b, R^r)$ is an ElGamal encryption of $z_b$*/
Randomly choose a message $m$
Randomly choose a witness $t \neq 1$
Simulate $V_1, V_2, V_3$ on the message $m$
Give $(m, A, B, t, V_1, V_2, V_3)$ to the adversary
Return $b'$: the output of the adversary
```

We first run the FINDER and obtain two members z_1 and z_2. Then a bit b is randomly chosen (out of our view) and we are given an encryption of z_b by ElGamal. Using the adversary through algorithm DISTINGUISHER, we can distinguish which one of z_1 or z_2 has been ElGamal encrypted, which is the break of semantic security.

5.3 Traceability and framing

The ability to open a group signature for the group manager is ensured by the correctness of V_3. Keep in mind that V_3 proves that the identity of the signer,

z, is correctly ElGamal encrypted. Anybody can thus be sure that the group leader would be able to open the signature if asked. Combined with V_1 and V_2, this proof ensures that the revealed member is a registered one: what is shown in these signature of knowledge is the knowledge of a membership certificate corresponding to the identity encrypted. Thus, avoiding traceability is at least as hard as the computation of an unregistered certificate or the break of the underlying signatures of knowledge.

The security against a framing attack is a bit more complicated. It can be stated as follows: no coalition of members nor the group leader can compute a valid group signature which, if opened, would be associated to somebody else.

Since the validity of a signature ensures that the signer knows a membership certificate (i.e. a solution to the equation $v^{e_2} = f_1 x^{e_1} + f_2$), a framing attack is hard if the following assumption holds:

Claim. No (adaptative) coalition can compute $k+1$ points on the curve $\mathcal{C}$: $Y^{e_2} = f_1 X^{e_1} + f_2$ when knowing only k points on it.

This assumption does not hold for every values of e_1 and e_2, f_1 and f_2. We now deal with what can be done to obtain an equivalent assumption, as well as the description of cases where the claim is false (which implies that a coalition attack is possible).

Case where $\mathbf{gcd}(e_1, e_2) = 1$ First, we can note that Claim 5.3 is equivalent to a simpler version in case that e_1 and e_2 are relatively prime; in that case, there exist λ and μ such that: $\lambda e_1 + \mu e_2 = 1$. Then the equation of $\mathcal{C}$ can be rewritten:

$$\begin{aligned} Y^{e_2} &= f_1^{\lambda e_1 + \mu e_2} X^{e_1} + f_2 \\ \left(\frac{Y}{f_1^{\mu}}\right)^{e_2} &= \left(f_1^{\lambda} X\right)^{e_1} + f_2 f_1^{-\mu e_2} \end{aligned}$$

or, by changing variables,

$$Y'^{e_2} = X'^{e_1} + d \ , \qquad \text{where} \ \ d = f_2 f_1^{-\mu e_2}$$

Thus, we just have to consider cases where $f_1 = 1$. Proving Claim 5.3 appears to be mathematically non-trivial, although it seems to be true.

Other cases If e_1 and e_2 are not relatively prime, a similar transformation can be performed, which modifies the values of the exponents. Let e be the greatest common divisor of e_1 and e_2, and note $e_1' = e_1/e$, $e_2' = e_2/e$. We now have $\gcd(e_1', e_2') = 1$ and we can write:

$$\begin{cases} Y' = Y^e f_1^{-\mu} \\ X' = X^e f_1^{\lambda} \end{cases}$$

$$Y'^{e_2'} = X'^{e_1'} + d' \ , \qquad \text{where} \ d' = f_2 f_1^{-\mu e_2'}$$

This does not appear to be interesting, because the transformation used is non-linear.

A framing attack when $e_1 = e_2$ If $e_1 = e_2$ the transformation proposed above is useless. However, if $f_1 = 1$, we can show that Claim 5.3 is false. Assuming that the common value $e = e_1 = e_2$ is small, it is possible for a coalition of 2^e registered members to compute a new membership certificate without the help of the group manager.

```
FRAMING($e$)
Choose a membership key $V_0$
$i \leftarrow 1$ , $k \leftarrow 2^e$
For $i \leftarrow 1$ to $k$
    $X_i \leftarrow V_{i-1}$
    $V_i \leftarrow$ REGISTER($X_i$)
Return $(V_k/2, X_1/2)$ as a new certificate
```

It easy to verify that such an algorithm produce new (unregistered) membership certificate. From k equations $V_i^e = X_i^e + f_2$ coming from registrations, we obtain by summation: $V_k = V_0^e + 2^e f_2$ and then:

$$\left(\frac{V_k}{2}\right)^e = \left(\frac{V_0}{2}\right)^e + f_2$$

This shows how a coalition of $k = 2^e$ members can forge a valid group signature which would be associated to an unexistent member if opened. Although such a problem can easily be avoided by carefully choosing group parameters, it is interesting to mention it as a new possible weakness of the scheme.

6 Conclusion

In this paper, we provide the first efficient solution to delete members from a group without compromising their past signatures or changing the group public key. The security of our mechanism is formally proven, as well as the underlying group signature scheme. However, obtaining members revocation with constant size signatures remains an open problem.

Acknowledgments

The authors especially thank the anonymous referees for helpful comments, including constructive remarks as well as minor corrections.

References

1. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A Practical and Provably Secure Coalition-Resistant Group Signature Scheme. In M. Bellare, editor, *Crypto '2000*, volume 1880 of *LNCS*, pages 255–270. Springer-Verlag, 2000.
2. G. Ateniese and G. Tsudik. Group Signature *à la carte*. In *10th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, January 1999.

3. G. Ateniese and G. Tsudik. Some Open Issues and New Directions in Group Signature. In *Financial Cryptography '99*, 1999.
4. M. Bellare and P. Rogaway. Random Oracles are Practical: a Paradigm for Designing Efficient Protocols. In *Proc. of the 1st Annual Conf. on Computer and Communications Security*. ACM Press, 1993.
5. J. Camenish and M.Michels. A Group Signature with Improved Efficiency. In K. Ohta and D. Pei, editors, *Asiacrypt '98*, volume 1514 of *LNCS*, pages 160–174. Springer-Verlag, 1999.
6. J. Camenish and M.Stadler. Efficient Group Signatures Schemes for Large Groups. In B. Kaliski, editor, *Crypto '97*, volume 1294 of *LNCS*, pages 410–424. Springer-Verlag, 1997.
7. R. Canetti and S. Goldwasser. An Efficient Threshold PKC Secure Against Adaptive CCA. In J. Stern, editor, *Eurocrypt '99*, volume 1592 of *LNCS*, pages 90–106. Springer-Verlag, 1999.
8. D. Chaum and E. van Heyst. Group Signatures. In D.W. Davies, editor, *Eurocrypt '91*, volume 547 of *LNCS*, pages 257–265. Springer-Verlag, 1992.
9. L. Chen and T.P. Pedersen. New Group Signature Schemes. In A. De Santis, editor, *Eurocrypt '94*, volume 950 of *LNCS*, pages 171–181. Springer-Verlag, 1995.
10. R. Cramer and V. Shoup. A Practical Public-Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack. In H. Krawczyk, editor, *Crypto '98*, volume 1462 of *LNCS*, pages 13–25. Springer-Verlag, 1998.
11. T. ElGamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In G.R. Blakley and D. Chaum, editors, *Crypto '84*, volume 196 of *LNCS*, pages 10–18. Springer-Verlag, 1985.
12. J. Kilian and E. Petrank. Identity Escrow. In H. Krawczyk, editor, *Crypto '98*, volume 1462 of *LNCS*, pages 169–185. Springer-Verlag, 1998.
13. S. Kim, S. Park, and D. Won. Convertible Group Signatures. In S. Kim and T. Matsumoto, editors, *Asiacrypt '96*, volume 1163 of *LNCS*, pages 311–321. Springer-Verlag, 1997.
14. S. Kim, S. Park, and D. Won. Group Signatures for Hierarchical Multigroups. In *Proc. of ISW '97*, volume 1396 of *LNCS*, pages 273–281. Springer-Verlag, 1998.
15. H. Petersen. How to Convert any Digital Signature Scheme into a Group Signature Scheme. In M. Lomas and S. Vaudenay, editors, *Proc. of Security Protocols Workshop '97*, volume 1361 of *LNCS*, pages 67–78. Springer-Verlag, 1997.
16. C. P. Schnorr. Efficient Identification and Signatures for Smart Cards. In G. Brassard, editor, *Crypto '89*, volume 435 of *LNCS*, pages 239–252. Springer-Verlag, 1990.