Quasi-Efficient Revocation of Group Signatures

Giuseppe Ateniese[1] and Gene Tsudik[2]

[1] Department of Computer Science
The Johns Hopkins University, Baltimore, MD. USA
ateniese@cs.jhu.edu
[2] Department of Information and Computer Science,
University of California, Irvine, CA. USA
gts@ics.uci.edu

Abstract. A group signature scheme allows any group member to sign on behalf of the group in an anonymous and unlinkable fashion. In the event of a dispute, a designated trusted entity can reveal the identity of the signer. Group signatures are claimed to have many useful applications such as voting and electronic cash.
A number of group signature schemes have been proposed to-date. However, in order for the whole group signature concept to become practical and credible, the problem of secure and efficient group member revocation must be addressed. In this paper, we construct a new revocation method for group signatures based on the signature scheme by Ateniese et al. [ACJT]. This new method represents an advance in the state-of-the-art since the only revocation schemes proposed thus far are: 1) based on implicit revocation and the use of fixed time periods, or 2) require the signature size to be linear in terms of the number of revoked members. Our method, in contrast, requires no time periods and offers constant-length signatures.

Keywords: Group signatures, revocation of group membership credentials, dynamic groups.

1 Introduction

Group signatures are a relatively new concept introduced by Chaum and van Heijst [CvH91] in 1991. A group signature, akin to its traditional counterpart, allows the signer to demonstrate knowledge of a secret with respect to a specific document. A group signature is publicly verifiable: it can be validated by anyone in possession of a group public key. However, group signatures are anonymous in that no one, with the exception of a designated group manager, can determine the identity of the signer. Furthermore, group signatures are unlinkable which makes it computationally hard to establish whether or not multiple signatures are produced by the same group member. In exceptional cases (such as a legal dispute) any group signature can be "opened" by a group manager to reveal unambiguously the identity of the actual signer. At the same time, no one — including the group manager — can misattribute a valid group signature.

These features of group signatures make them attractive for many specialized applications, such as voting and bidding. They can, for example, be used in invitations to submit tenders [CP95]. All companies submitting a tender form a group and each company signs its tender anonymously using the group signature. Once the preferred tender is selected, the winner can be traced while the other bidders remain anonymous.

More generally, group signatures can be used to conceal organizational structures, e.g,, when a company or a government agency issues a signed statement. Group signatures can also be integrated with an electronic cash system whereby several banks can securely distribute anonymous and untraceable e-cash. The group property can offer a further advantage of concealing the identity of the cash-issuing bank [LR98].

Several interesting group signature schemes have been recently proposed [CP95,CvH91,CS97,AT99a,CM98a,Cam97]. Some have been subsequently broken, others are impractical due to long public keys and/or long signatures while most remaining schemes offer uncertain (i.e., unproven) security. One exception is a recent group signature scheme by Ateniese et al. [ACJT] which is referred to as ACJT from here on. This scheme is particularly attractive since it is both efficient and provably secure.

Motivation As observed in [AT99], to be truly useful, a group signature scheme must support dynamic group membership. Current state-of-the-art group signature schemes (such as Camenisch/Stadler [CS97], Camenisch/Michels

[CM98a], ACJT [ACJT]) support growing membership: new members can join without precipitating changes in the group public key or re-issuing group membership certificates for existing members. However, *shrinking* group membership has not been given the same attention. We believe this is because either it was not deemed important enough, or (more likely) no viable solutions were known.

In many realistic group settings, group members are equally likely to join, leave voluntarily or be excluded, from the group. Therefore, we consider supporting growing and shrinking membership of equal importance. Starting from this premise, we claim that group signature schemes, no matter how elegant or how secure, will remain a neat and curious tool in a theoretical cryptographer's "bag-of-tricks" until a secure and efficient method to support both growing and shrinking membership is found.

Contribution In this paper, we construct and demonstrate an effective and secure revocation method for group signatures based on the signature scheme by Ateniese et al. [ACJT]. This new method represents an advance in the state-of-the-art since the only revocation schemes proposed thus far are either:[1]

1. Based on implicit revocation, (loosely) synchronized clocks and the use of fixed time periods, or
2. Require group signature size to be $O(n)$ where n is the number of revoked members and ask the signer to perform $O(n)$ work to compute each signature.

Our method, in contrast, offer explicit (CRL-based) revocation, requires no time periods and offers constant-length signatures and constant work for signers.

Broadly speaking, this paper has but one contribution: it demonstrates the first viable group revocation scheme based on the only provably secure and efficient group signature scheme proposed to-date. At the same time, it should be noted from the outset that the revocation method described in this paper is – albeit viable – is not quite practical for reasons to be discussed below. However, we believe that this result moves the field one (perhaps, small) step closer to reality.

Organization The rest of the paper is organized as follows. We first provide, in the next section, an overview of related work. In Section 3 we discuss some preliminaries. Next, Section 4 discusses revocation issues in the context of group signatures. Section 5, summarizes the Ateniese et al. group signature scheme. Sections 6 overviews a very simple revocation scheme followed by the new quasi-efficient revocation scheme and its informal analysis presented in Sections 7 and 8, respectively. We conclude the paper with the summary and future work.

2 Related Work

Last year, Bresson and Stern [BS2000] proposed the first viable and elegant solution for revocation of group signatures. Unfortunately, their solution requires the signature size to be linear with respect to the number of revoked members. Moreover, it is based on the group signature scheme proposed by Camenisch and Stadler which has been found later to have certain security problems. (It should be noted that, even in its corrected/modified version, this scheme has not been proven secure, as it relies on two non-standard assumptions.)

In a very recent paper, Song [Song] proposed two interesting revocation methods based, like the present work, on the ACJT scheme. (Recall that ACJT is provably secure.) Both methods are notable since – in addition to standard revocation – they also provide retroactive revocation as well as forward security. (In fact, the emphasis is on forward security.) Moreover, they offer constant-length signatures which is an improvement over the Bresson and Stern's result. However, one important feature of Song's methods is the the use of fixed (in both length and number) time periods to support revocation. In particular, each member's certificate must evolve in every time period and any and all verifiers must be aware of this evolution. Also, the maximum number of time periods is fixed and embedded in each member's group certificate. While appropriate for some settings, this solution is not very general since it is hard (in fact, impossible) to revoke a member within a time period. Furthermore, the security of one of the methods is based on a new and perhaps uncertain cryptographic assumption which is appreciably stronger than the Decision Diffie-Hellman (DDH) assumption. The second scheme relies on the existence of an efficient method (one-way function) of deterministically computing a fixed-length sequence of prime numbers starting with an initial prime. It remains to be seen whether practical examples of this method are possible.

[1] See Section 2 for details.

3 Preliminaries

Group-signature schemes are typically defined as follows:[2]

Definition 1. A *group signature scheme* is a digital signature scheme comprised of the following five procedures:

SETUP: A probabilistic algorithm which – on input of a security parameter ℓ – outputs the initial group public key $\mathcal{Y}$ (including all system parameters) and the secret key $\mathcal{S}$ for the group manager.

JOIN: A protocol between the group manager and a user that results in the user becoming a new group member. The user's output is a membership certificate and a membership secret.

SIGN: A probabilistic algorithm that on input a group public key, a membership certificate, a membership secret, and a message m outputs group signature of m.

VERIFY: An algorithm for establishing the validity of an alleged group signature of a message with respect to a group public key.

OPEN: An algorithm that, given a message, a valid group signature on it, a group public key and a group manager's secret key, determines the identity of the signer.

A secure group signature scheme must satisfy the following properties:

Correctness: Signatures produced by a group member using SIGN must be accepted by VERIFY.

Unforgeability: Only group members are able to sign messages on behalf of the group.

Anonymity: Given a valid signature of some message, identifying the actual signer is computationally hard for everyone but the group manager.

Unlinkability: Deciding whether two different valid signatures were computed by the same group member is computationally hard.

Exculpability: Neither a group member nor the group manager can sign on behalf of other group members.[3]

Traceability: The group manager is always able to open a valid signature and identify the actual signer. Therefore, any colluding subset of group members cannot generate a valid signature that the group manager cannot link to one of the colluding group members.

In order to provide revocation of membership, an additional property is necessary:

Revocability: A signature produced using SIGN by a revoked member must be rejected using a (potentially modified) VERIFY. Equivalently, a signature produced using SIGN by a valid member must be accepted by VERIFY.

The efficiency of of a group signature scheme depends on a number of factors. Usually, the costs of SIGN and VERIFY as well as the sizes of the group signature and the group public key are the most important efficiency measures.

4 Revocation Preliminaries

In general, as soon as a member (Alice, as usual) is revoked, there must be a way to unambiguously determine her revocation status. At the same time, all signatures generated by Alice before revocation must remain valid and secure, i.e., anonymous and unlinkable. This property was first defined in [AT99] and later referred (and refined) by Song [Song] as *backward unlinkability*. There are, of course, exceptions to the above. For example, if revocation takes place for reasons of fraud, it might be necessary to link and identify **all** signatures generated by Alice.

One simple way to obtain revocation is to issue a new group public key and new group certificates to all valid members whenever a group member (or a number thereof) leaves or is ejected. However, this would entail a heavy cost and a significant inconvenience. First, all potential verifiers must be notified of the change. This appears to be unavoidable. Second, all remaining members must participate in a JOIN protocol with the group manager. This represents an extra burden for the members since the JOIN protocol is always on-line and involves a lot computation (as compared to SIGN or VERIFY).

---
[2] An in-depth discussion on this subject can be found in [Cam98].

[3] However, nothing precludes the group manager from creating phantom signers and then producing group signatures. The same risk occurs with respect to CA-s in traditional (non-group) PKI-s.

However, it is possible to avoid running interactive JOIN protocols with all members. This can be achieved by generating a new group public key and issuing new group certificates for all members **without** any interaction. As an illustration, we sketch out (in Section 6) a simple method based on the ACJT scheme. (A very similar approach can be constructed with the Camenisch/Stadler group signature scheme [CS97].) However, this approach is not very practical as it involves the issuance of many new membership certificates and requires each group member to fetch its new certificate following every member leave event.

To achieve more effective and efficient revocation, we need to avoid issuing new group certificates to non-revoked members. An ideal revocation method would employ the revocation paradigm commonly used in traditional signature schemes: a verifier simply checks the signer's certificate against the current Certificate Revocation List (CRL). This paradigm is attractive since the signer is unaware of the ever-changing CRL and the revocation checking burden is placed on the verifier. In our setting, however, a group signature always contains in some form an encrypted version of the signer's group certificate. As pointed out in [ACJT], encryption of the certificate must be semantically secure in order to prevent linking of group signatures.

The very same semantic security makes it impossible for the verifier to link a group signature to a (potentially) revoked group certificate (or some function thereof) that has to appear as part of a CRL. To see why this is the case, consider the opposite: if a verifier is able to link a single group signature to a certain CRL entry, then the same verifier can link multiple group signatures (all by one signer) to the very same CRL entry. This is not a problem if the signer is revoked before all of these group signatures are generated. However, if a verifier can link (based on a current CRL) a revoked signer's signatures computed before revocation, the property of *backward unlinkability* is not preserved. Therefore, we claim that the signer must somehow factor in the current CRL when generating a group signature. In fact, the signer must prove, as part of the signing, that its group certificate (or a function thereof) is not part of the current CRL.

The above is the general approach we take in this paper. The method outlined in detail below (in Section 7) requires a signer to prove non-membership of the current CRL as part of signature generation. The verifier, in turn, checks revocation as part of signature verification. The end-result is that the notion of the group public key is extended to include the latest group CRL.

Revocation Efficiency We identify the following measures of efficiency or practicality for any revocation method (of course, only in the context of group signatures):

– **Increased Signature Size**: is the most important measure of a revocation method's efficiency. Clearly, signature size should be minimized. More generally, if the underlying group signature scheme has $O(x)$-size signatures (where x is a constant, or some function of group size), revocation checking should ideally not change the signature size in the $O()$ notation.
– **Signer Cost:** is the additional cost of generating a group signature that also proves non-revocation of the signer. Ideally, this added cost is constant.
– **Verifier Cost:** is the additional cost of verifying a group signature that also proves non-revocation of the signer. As above, this added cost is, at best, constant.
– **CRL Size:** is an essential measure since it, in fact, determines the effective overall size of the group public key.
– **CRL Issuance Cost:** is the cost of composing and issuing a new CRL (by the group manager) each time a group member must be revoked. While not completely negligible, this efficiency measure is the least significant of the above.

5 The ACJT Group Signature Scheme

In this section we provide an overview of the ACJT scheme [ACJT]. (Readers familiar with ACJT may skip this section with no loss of continuity.) In its interactive, identity escrow form, the ACJT scheme is proven secure and coalition-resistant under the Strong RSA and DDH assumptions. The security of the non-interactive group signature scheme relies additionally on the Fiat-Shamir heuristic (also known as the random oracle model.

Let $\epsilon > 1$, k, and ℓ_p be security parameters and let λ_1, λ_2, γ_1, and γ_2 denote lengths satisfying $\lambda_1 > \epsilon(\lambda_2+k)+2$, $\lambda_2 > 4\ell_p$, $\gamma_1 > \epsilon(\gamma_2 + k) + 2$, and $\gamma_2 > \lambda_1 + 2$. Define the integral ranges $\Lambda =]2^{\lambda_1} - 2^{\lambda_2}, 2^{\lambda_1} + 2^{\lambda_2}[$ and $\Gamma =]2^{\gamma_1} - 2^{\gamma_2}, 2^{\gamma_1} + 2^{\gamma_2}[$. Finally, let $\mathcal{H}$ be a collision-resistant hash function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^k$. (The parameter ϵ controls the tightness of the statistical zero-knowledge and the parameter ℓ_p sets the size of the modulus to use.)

In the initial phase, the group manager (GM) sets the group public key $\mathcal{Y}$ as well as its own secret key $\mathcal{S}$.

SETUP:

1. Select random secret ℓ_p-bit primes p', q' such that $p = 2p' + 1$ and $q = 2q' + 1$ are prime. Set the modulus $n = pq$.
2. Choose random elements $a, a_0, g, h \in_R \mathrm{QR}(n)$ (of order $p'q'$).
3. Choose a random secret element $x \in_R \mathbb{Z}^*_{p'q'}$ and set $y = g^x \bmod n$.
4. The group public key is: $\mathcal{Y} = (n, a, a_0, y, g, h)$.
5. The corresponding secret key (known only to GM) is: $\mathcal{S} = (p', q', x)$.

Suppose now that a new user wants to join the group. We assume that communication between the user and the group manager is secure, i.e., private and authentic. The selection of per-user parameters is done as follows:

JOIN:

1. User P_i generates a secret exponent $\tilde{x}_i \in_R]0, 2^{\lambda_2}[$, a random integer $\tilde{r} \in_R]0, n^2[$ and sends $C_1 = g^{\tilde{x}_i} h^{\tilde{r}} \bmod n$ to GM and proves him knowledge of the representation of C_1 w.r.t. bases g and h.
2. GM checks that $C_1 \in \mathrm{QR}(n)$. If this is the case, GM selects α_i and $\beta_i \in_R]0, 2^{\lambda_2}[$ at random and sends (α_i, β_i) to P_i.
3. User P_i computes $x_i = 2^{\lambda_1} + (\alpha_i \, \tilde{x}_i + \beta_i \bmod 2^{\lambda_2})$ and sends GM the value $C_2 = a^{x_i} \bmod n$. The user also proves to GM:
   (a) that the discrete log of C_2 w.r.t. base a lies in Λ, and
   (b) knowledge of integers u, v, and w such that
      i. u lies in $] - 2^{\lambda_2}, 2^{\lambda_2}[$,
      ii. u equals the discrete log of $C_2/a^{2^{\lambda_1}}$ w.r.t. base a, and
      iii. $C_1^{\alpha_i} g^{\beta_i}$ equals $g^u (g^{2^{\lambda_2}})^v h^w$

   (The statements (i–iii) prove that the user's membership secret $x_i = \log_a C_2$ is correctly computed from C_1, α_i, and β_i.)
4. GM checks that $C_2 \in \mathrm{QR}(n)$. If this is the case and all the above proofs were correct, GM selects a random prime $e_i \in_R \Gamma$ and computes $A_i := (C_2 \, a_0)^{1/e_i} \bmod n$. Finally, GM sends P_i the new membership certificate $[A_i, e_i]$. (Note that $A_i = (a^{x_i} \, a_0)^{1/e_i} \bmod n$.)
5. User P_i verifies that $a^{x_i} a_0 \equiv {A_i}^{e_i} \pmod{n}$.

Thereafter, the new member can generate group signatures as follows:

SIGN:

1. Generate a random value $w \in_R \{0,1\}^{2\ell_p}$ and compute:

$$T_1 = A_i \, y^w \bmod n, \quad T_2 = g^w \bmod n, \quad T_3 = g^{e_i} \, h^w \bmod n \ .$$

2. Randomly choose $r_1 \in_R \pm\{0,1\}^{\epsilon(\gamma_2+k)}$, $r_2 \in_R \pm\{0,1\}^{\epsilon(\lambda_2+k)}$, $r_3 \in_R \pm\{0,1\}^{\epsilon(\gamma_1+2\ell_p+k+1)}$, and $r_4 \in_R \pm\{0,1\}^{\epsilon(2\ell_p+k)}$ and compute:
   (a) $d_1 = {T_1}^{r_1}/(a^{r_2} \, y^{r_3}) \bmod n$, $d_2 = {T_2}^{r_1}/g^{r_3} \bmod n$, $d_3 = g^{r_4} \bmod n$, and $d_4 = g^{r_1} \, h^{r_4} \bmod n$;
   (b) $c = \mathcal{H}(g \| h \| y \| a_0 \| a \| T_1 \| T_2 \| T_3 \| d_1 \| d_2 \| d_3 \| d_4 \| m)$;
   (c) $s_1 = r_1 - c(e_i - 2^{\gamma_1})$, $s_2 = r_2 - c(x_i - 2^{\lambda_1})$, $s_3 = r_3 - c \, e_i \, w$, and $s_4 = r_4 - c \, w$ (all in $\mathbb{Z}$).
3. Output $(c, s_1, s_2, s_3, s_4, T_1, T_2, T_3)$.

A group signature is basically a signature of knowledge of (1) a value $x_i \in \Lambda$ such that $a^{x_i} a_0$ is the value that is ElGamal-encrypted in (T_1, T_2) under y and of (2) an e_i-th root of that encrypted value, where e_i is the first part of the representation of T_3 w.r.t. g and h and that e_i lies in Γ.

A verifier checks the validity of a signature $(c, s_1, s_2, s_3, s_4, T_1, T_2, T_3)$ on a message m as follows:

VERIFY:

1. Compute:

$$c' = \mathcal{H}\big(g\|h\|y\|a_0\|a\|T_1\|T_2\|T_3\|{a_0}^c\,{T_1}^{s_1-c2^{\gamma_1}}/(a^{s_2-c2^{\lambda_1}}\,y^{s_3}) \bmod n\,\| \\ {T_2}^{s_1-c2^{\gamma_1}}/g^{s_3} \bmod n\,\|{T_2}^c\,g^{s_4} \bmod n\,\|{T_3}^c\,g^{s_1-c2^{\gamma_1}}\,h^{s_4} \bmod n\,\|\,m\big)\ .$$

2. Accept the signature if and only if $c = c'$, and $s_1 \in \pm\{0,1\}^{\epsilon(\gamma_2+k)+1}$, $s_2 \in \pm\{0,1\}^{\epsilon(\lambda_2+k)+1}$, $s_3 \in \pm\{0,1\}^{\epsilon(\lambda_1+2\ell_p+k+1)+1}$, $s_4 \in \pm\{0,1\}^{\epsilon(2\ell_p+k)+1}$.

In the event that the signer must be subsequently identified (e.g., in case of a dispute) GM executes the following procedure:

OPEN:

1. Check the signature's validity via the VERIFY procedure.
2. Recover A_i (and thus the identity of P_i) as $A_i = T_1/{T_2}^x \bmod n$.
3. Prove that $\log_g y = \log_{T_2}(T_1/A_i \bmod n)$

6 Simple Revocation in ACJT

Recall that the JOIN protocol in the ACJT scheme results in the issuance of a secret membership certificate $[A_i, e_i]$ where $A_i = (a^{x_i}\,a_0)^{1/e_i} \bmod n$.

Since the group manager is the one choosing each prime e_i at JOIN time, it can easily issue a new certificate to every valid group member without any additional interaction. Specifically, GM can issue a new certificate of the form:

$$A_{k,i} = (a_k^{x_i}\,a_{0,k})^{1/e_i} \bmod n,\ e_i$$

We use the index k to denote the sequence number of U_i's group certificate; equivalently, k is the number of shrinking membership changes that took place since the U_i joined the group.

The values a_k and $a_{0,k}$ are generated by GM for every update (re-issue) of the group public key. One simple and efficient way of generating these values for GM to select a secret random number $r \in \mathbb{Z}_{p'q'}$ and compute $a_k = a_{k-1}^r \bmod n$ and $a_{0,k} = a_{0,k-1}^r \bmod n$. With overwhelming probability a_k and $a_{0,k}$ will also have order $p'q'$. Notice that a revoked user can not obtain a new-issue group certificate since the value r is not known. Furthermore, GM can easily compute the value $a_k^{x_i}$ as $a_k^{x_i} = (a_{k-1}^{x_i})^r$ (the initial value of $a_{k-1}^{x_i}$ can be stored by GM during JOIN).

Next, GM publishes all newly issued certificates in some public forum, e.g., a bulletin board or a web page. Alternatively, it can broadcast the whole batch to the group. Of course, to keep the number of group members secret, GM can (and should) also issue and publish a sufficient number of fake certificates. The new certificates are accompanied by a new group public key:

$$\mathcal{Y}_k = (n, a_k, a_{0,k}, y, g, h)$$

Obviously, group certificates can not be published in cleartext. Instead, each certificate must be individually encrypted and tagged. One possible format is illustrated in Table 6. The purpose of the search tag is to help each member find its new certificate in the table. (Otherwise, a member would have to try decrypting $n/2$ certificates, on the average, before finding its own.) In this example, every new certificate is encrypted (e.g., using Cramer/Shoup [CS98] under a public key provided by each user at JOIN time). A search tag can be computed, for example, as a one-way function (a cryptographically strong hash function such as SHA would suffice) of each user's previous membership certificate $A_{k-1,i}$. Other ways to compute tags are possible. For example, a pair-wise secret can be established between GM and every member during JOIN. Some function of that secret can serve as a search tag.

The present method is both simple and secure. Unfortunately, it is inefficient, since – for every leaving or expelled member – GM needs to perform $O(n)$ cryptographic operations to compose the table. Moreover, each member

Encrypted Certificate	Search Tag
$E_1(A_{k,i}, e_i)$	$F(A_{k-1,1})$
...	...
...	...
...	...
$E_n(A_{k,n}, e_n)$	$F(A_{k-1,i})$

Table 1. Re-issued Group Certificate Table

needs to fetch the entire certificate table (containing its new certificate) as well as the new group public key. Note that just fetching one's own certificate is insecure as it would reveal to a potential eavesdropper the ownership of a group certificate.

7 More Effective Revocation

We begin by assuming, as usual, that a CRL is a structure available at all times from a number of well-known public repositories or servers. A CRL is also assumed to be signed and timestamped by its issuer which can be a universally trusted CA, a group manager or some other trusted party.

In addition to the usual components of a group signature scheme (`SETUP, JOIN`, etc.) we introduce an additional algorithm called `REVOKE`. Also, as can be expected, revocation influences `SIGN` and `VERIFY` algorithms. The `JOIN` and `OPEN` components remain unchanged. The only (addition) change in `SETUP` is as follows:

`SETUP (new step):`

– Select $\bar{G} =< \bar{g} >$ of order n in which computing discrete logarithms is hard. For example, $\bar{G}$ can be a subgroup of $Z^*_{\bar{p}}$ for a prime $\bar{p}$ such that n divides $(\bar{p}-1)$.

The new `REVOKE` algorithm shown below is executed by the group manager whenever a member (or a collection of members) leaves or is expelled. (Note that `REVOKE` may also be executed as a "decoy", i.e., without any new membership revocation activity.) The cost to the GM is linear in the number of revoked group members.

`REVOKE:` (We use s to denote the index of the current CRL issue.)

1. First, choose a random element $b_s \in_R \mathrm{QR}(n)$ (of order $p'q'$). b_s becomes the current revocation base.
2. WLOG, assume that m users: $U_1, ..., U_m$ are to be revoked.
3. For each revoked $U_j,\ 0 \geq\ j \leq\ m$ compute:

$$V_{s,j} = b_s^{e_j}$$

4. The actual revocation list is then published:

$$CRL_s = [b_s, V_{s,j} \mid 0 < j < m+1]$$

In the amended `SIGN` algorithm, as part of step 1, member U_i generates two additional values:

$$T_4 = f = \bar{g}^r \text{ where } r \in_R Z^*_n$$

$$T_5 = f^{b_s^{e_i}} \bmod n$$

U_i then proves, in zero knowledge, that the double discrete logarithm of T_4 with bases f and b_s, respectively is the same as the discrete logarithm of T_3's representation base $\bar{g}$ and h respectively. Since T_3 is computed as $g^{e_i}\, h^w \bmod n$, the resulting proof of knowledge (SKLOGEQLOGLOG) is verifiable if and only if the same e_i is used the construction of both T_4 and T_3. The details of this proof are presented in Section 7 below.

Remark: the current CRL, CRL_s must be signed as part of the message "m" which serves as input to the hash function in the actual signature-of-knowledge. This commits the signer to a specific CRL epoch.

In the amended `VERIFY` algorithm we introduce a new steps 3 and 4:

3. For each $V_{s,j} \in CRL$, check if:
$$T_5 \ == \ T_4^{V_{s,j}} \bmod n$$
4. Check SKLOGEQLOGLOG, the proof of equality of double discrete logarithm of T_5 and discrete logarithm of T_3's representation base $\bar{g}$.

The intuition behind this scheme is straight-forward: if a member U_i is revoked, $V_{s,i}$ is published as part of the current group CRL. Thereafter, in order to produce a group signature, U_i needs to prove that $(b_s)^{e_i}$ does not appear on the CRL which is impossible since $(b_s)^{e_i} = V_{s,j}$ for some j if U_i is revoked.

We claim that the new scheme provides *backward unlinkability* because signatures produced by a revoked user prior to revocation in earlier CRL *epochs* can not be linked to those produced after revocation. Suppose that an adversary is able to link a pre-revocation signature to a post-revocation signature. Then, she can only do so with the help of the *new* values: T_4 and T_5. (Otherwise the ACJT scheme is insecure). Since $T_4 = f$ is chosen at random for each signature, the only way the adversary can link two signatures is using $T_5 = f^{b_s^{e_i}}$. However, this is impossible since the respective b_s values are different and unrelated for any pair of signatures computed in different CRL epochs.

To be more specific, we need to consider two cases: linking two signatures from different CRL epochs and linking two signatures from the same CRL epoch. It is easy to see that the former is infeasible for some $T_5^1 = f'^{b_s'^{e_i}}$ and $T_5^2 = f"^{b_s"^{e_i}}$ where $f' \neq f"$ and $b_s' \neq b_s"$. The latter is also infeasible for some $T_5^1 = f'^{b_s^{e_i}}$ and $T_5^2 = f"^{b_s^{e_i}}$ where $f' \neq f"$, based on a well-known variant of the DDH problem.

Obscuring CRL Size. Over time, the size of the CRL may leak some information about the population of the group: by observing long-term changes and fluctuations in the size of the CRL, an adversary can guess the number of group members. For this reason, it may be necessary to obscure the true CRL size. This can be done by introducing a number of fake (but well-formed) CRL entries.

Proofs Involving Double Discrete Logs Proofs of knowledge of double discrete logarithms have been used in the past. Examples include Stadler's technique for publicly verifiable secret sharing [Stad96] and Camenisch/Stadler group signature scheme and its derivatives [CS97,LR98,KP98]. All of the these involve only proofs of knowledge; whereas, in the above scheme, we need a new proof (SKLOGEQLOGLOG) of equality of a double discrete log and a (single) discrete log of a representation. The technique we use is a minor variation of the SKLOGLOG (proof of knowledge of double discrete logarithm) proposed by Camenisch [Cam98], Stadler [Stad96] and Camenisch/Stadler [CS97]. The proof is constructed as follows:

Given $y_1 = g^{a^x}$ and $y_2 = g_1^x g_2^w$, we want to prove that:
$$Dlog_a(Dlog_g y_1) = Dlog_{g_1}(y_2/g_2^w) \ \ (= x)$$
Let $\ell \leq k$ be two security parameters and $H : \{0,1\}^* \longrightarrow \{0,1,\}^k$ be a cryptographic hash function. Generate 2ℓ random numbers $r_1, \ldots, r_\ell$ and $v_1, \ldots, v_\ell$. Compute, for $1 \leq i \leq \ell$, $t_i = g^{a^{r_i}}$ and $t_i' = g_1^{r_i} g_2^{v_i}$. The signature of knowledge on the message m is $(c, s_1, s_2, \ldots, s_\ell, s_1', s_2', \ldots, s_\ell')$, where:
$$c = H(m||y_1||y_2||g||a||||g_1||g_2||t_1||\cdots||t_\ell||t_1'||\cdots||t_\ell')$$
and
$$\textbf{if } c[i] = 0 \textbf{ then } s_i = r_i, s_i' = v_i;$$
$$\textbf{else } s_i = r_i - x, s_i' = v_i - w;$$
To verify the signature it is sufficient to compute:
$$c' = H(m||y_1||y_2||g||a||g_1||g_2||\bar{t}_1||\cdots||\bar{t}_\ell||\bar{t}_1'||\cdots||\bar{t}_\ell')$$
with

$$\textbf{if } c[i]=0 \textbf{ then } \bar{t}_i = g^{a^{s_i}}, \bar{t}'_i = g_1^{s_i} g_2^{s'_i};$$

$$\textbf{else } \bar{t}_i = y^{a^{s_i}}, \bar{t}'_i = y_2 g_1^{s_i} g_2^{s'_i};$$

and check whether $c = c'$.

8 Efficiency Considerations

The new revocation scheme presented in Section 7 is quasi-efficient in that a group signature is of **fixed size** and a signer performs a **constant** amount of work in generating a signature. This is, as claimed earlier, an improvement on prior results. However, proofs involving double discrete logs are notoriously expensive. For example, if we assume a hash function $H : \{0,1\}^* \longrightarrow \{0,1,\}^k$ where $k = 160$ bits (as in SHA-1), and we assume that the security parameter $\ell = k$, then each `SIGN` operation will take approximately 500 exponentiations. The cost of `VERIFY` is roughly the same. Moreover, with a 1024-bit modulus, a signature can range into hundreds of Kbits. This is clearly not efficient.

Remark: one way to reduce the costs of `SIGN` and `VERIFY` and (roughly) halve the number of exponentiations is to ask the signer to release as part of `SIGN` two additional values: $T_6 = \hat{g}$ and $T_7 = \hat{g}^{e_i}$ for a randomly chosen $\hat{g} \in QR_n$. The signer would then prove the equality of the discrete log base $\hat{g}$ of T_7 and the double discrete log of T_5 (base f and b_s, respectively). This proof would be both shorter and less costly then the proof of equality of double discrete log (of T_5) and discrete log of representation of T_3.

Despite the usage of double discrete logarithm proofs, in contrast with Bresson and Stern's scheme [BS2000], the cost of `SIGN` in our scheme is constant (independent of group size of number of revoked members) and signatures are of a fixed size. Comparing with Song's schemes, our scheme is more expensive for both `SIGN` and `VERIFY` due to the double discrete log proof. One advantage of our scheme is in not using fixed (in length and number) time periods. Consequently, a new revocation list can be issued at any time. Also, we introduce no new cryptographic assumptions. Song's two schemes, however, have the benefit of *retroactive public revocability* meaning that a member's signatures can be revoked for one or more **past** time periods. This is a feature that our method does not possess.

The the cost of `REVOKE` in our scheme is linear in the number of revoked members: GM performs one exponentiation for CRL entry $V_{s,j}$. This is comparable with prior results in both [BS2000] and [Song] schemes.

9 Summary and Future Work

In this paper, we presented a new revocation method for group signatures based on the ACJT signature scheme [ACJT]. The new method is more practical than prior art due to fixed-size signatures and constant work by signers. On the other hand, it requires the use of proofs-of-knowledge involving double discrete logs which results in hundreds of exponentiations per signature. Consequently, revocation in group signatures remains inefficient while the following issues remain open:

- Shorter CRL: in our method a CRL is proportional to the number of revoked members. An ideal scheme would have a fixed-size or, at least, a shorter CRL (e.g., logarithmic in the number of revoked members).
- More efficient VERIFY: the cost of VERIFY is linear in the number of revoked members. It remains to be seen whether a constant- or sublinear-cost VERIFY can be devised.
- Double discrete log: proofs using double discrete logarithms are inefficient, requiring many exponentiations. For revocation to become truly practical, we need to devise either more efficient double discrete log proofs or different revocation structures that avoid double discrete log proofs altogether.

References

[ACJT] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A Practical & Provably Secure Coalition-Resistant Group Signature Scheme In *Advances in Cryptology — CRYPTO '00*, Springer-Verlag, 2000.

[CP95] L. Chen and T. P. Pedersen. New group signature schemes. In *Advances in Cryptology — EUROCRYPT '94*, vol. 950 of *LNCS*, pp. 171–181, 1995.

[CS97] J. Camenisch and M. Stadler. Efficient group signature schemes for large groups. In *Advances in Cryptology — CRYPTO '97*, vol. 1296 of *LNCS*, pp. 410–424, Springer-Verlag, 1997.

[Stad96] M. Stadler. Publicly Verifiable Secret Sharing, In *Advances in Cryptology — EUROCRYPT '96*, Springer-Verlag, 1996.

[Cam98] J. Camenisch. Group signature schemes and payment systems based on the discrete logarithm problem. PhD thesis, vol. 2 of *ETH Series in Information Security an Cryptography*, Hartung-Gorre Verlag, Konstanz, 1998. ISBN 3-89649-286-1.

[CvH91] D. Chaum and E. van Heyst. Group signatures. In *Advances in Cryptology — EUROCRYPT '91*, vol. 547 of *LNCS*, pp. 257–265, Springer-Verlag, 1991.

[KP98] J. Kilian and E. Petrank. Identity escrow. In *Advances in Cryptology — CRYPTO '98*, vol. 1642 of *LNCS*, pp. 169–185, Springer-Verlag, 1998.

[LR98] A. Lysyanskaya and Z. Ramzan. Group blind digital signatures: A scalable solution to electronic cash. In *Financial Cryptography (FC '98)*, vol. 1465 of *LNCS*, pp. 184–197, Springer-Verlag, 1998.

[CS98] R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Advances in Cryptology — CRYPTO '98*, vol. 1642 of *LNCS*, pp. 13—25.

[AT99a] G. Ateniese and G. Tsudik. Group Signatures a' la Carte. In Proceedings of 1999 ACM Symposium on Discrete Algorithms, ACM Press, 1999.

[AT99] G. Ateniese and G. Tsudik. Some open issues and new direction in group signatures. In Proceedings of 1999 Financial Crypto. Springer-Verlag, 1999.

[BS2000] E. Bresson and J. Stern. Efficient Revocation in Group Signatures, In Proceedings of Public Key Cryptography (PKC'2001), Springer-Verlag, 2001.

[Song] D. Song. Practical Forward-Secure Group Signature Schemes. In Proceedings of 2001 ACM Symposium on Computer and Communication Security. November 2001.

[Cam97] J. Camenisch. Efficient and Generalized Group Signatures. In Advances in Cryptology - EUROCRYPT '97, vol. 1233 of *LNCS*, pp. 465-479, Springer Verlag, 1997

[CM98a] J. Camenisch and M. Michels. A group signature scheme with improved efficiency. In *Advances in Cryptology — ASIACRYPT '98*, vol. 1514 of *LNCS*, pp. 160–174, Springer-Verlag, 1998.