Verifier-Local Revocation Group Signature Schemes with Backward Unlinkability from Bilinear Maps

Toru Nakanishi and Nobuo Funabiki

Department of Communication Network Engineering,
Okayama University, 3-1-1 Tsushima-Naka,
Okayama 700-8530, Japan
{nakanisi, funabiki}@cne.okayama-u.ac.jp

Abstract. An approach of membership revocation in group signatures is verifier-local revocation (VLR for short). In this approach, only verifiers are involved in the revocation mechanism, while signers have no involvement. Thus, since signers have no load, this approach is suitable for mobile environments. Although Boneh and Shacham recently proposed a VLR group signature scheme from bilinear maps, this scheme does not satisfy the backward unlikability. The backward unlikability means that even after a member is revoked, signatures produced by the member before the revocation remain anonymous. In this paper, we propose VLR group signature schemes with the backward unlinkability from bilinear maps.

1 Introduction

A *group signature scheme* [10,8,1,15,2,9,16,13,6,7,14] allows a group member to anonymously sign a message on behalf of a group, where a group manager controls the membership of members. Then, often a third party can cancel the anonymity of signatures to trace the signers. Some schemes support membership revocation [15,2,9,16,6,7], where the membership of a member can be disabled without influencing the other members.

This paper focuses on the membership revocation. The simplest revocation method is that the manager changes the group public key and secret keys of all members except the revoked member to re-distribute the keys [2]. However, the other members' loads are enormous. A better solution is to broadcast a small public membership message to all signers and verifiers, as in [9,16,6]. Although the costs of signers are better, the signer still has to obtain some data depending on the size of the group (or the number of revoked members) whenever signing. On the other hand, there is another approach [15,2,7], where some revocation messages are only sent to verifiers, although the verifiers need the computational cost depending the number of revoked members. Since the signers' costs are lower, this type is suitable for mobile environments where mobile hosts anonymously communicate with the servers. We refer to this type as *Verifier-Local Revocation* (VLR for short) group signature scheme, as in [7].

B. Roy (Ed.): ASIACRYPT 2005, LNCS 3788, pp. 533–548, 2005.

In [15,2], VLR group signature schemes based on the strong RSA assumption are proposed. However, the schemes have some drawbacks on efficiency. The first scheme of [15] and the scheme of [2] suffer from the inefficiency of signing, due to the used inefficient zero-knowledge proofs. The second scheme of [15] forces a signer to compute $O(T)$ exponentiations at every time interval, where T is the total number of time intervals. Since the revocation can be performed only at the beginning of each time interval, T should be large. This means the signer's heavy load.

In [7], a VLR group signature scheme based on bilinear maps is proposed by Boneh and Shacham. The advantage of this scheme is that signatures are short, since the elliptic curves can be adopted. On the other hand, the schemes of [15,2] have an advantage over [7], *backward unlinkability.* This property means that even after a member is revoked, signatures produced by the member before the revocation remain anonymous. However, in the scheme of [7], all the signatures produced from the revoked member are linkable. This means that the anonymity of signatures produced before the revocation is compromised. In some cases that all signatures from an illegal person should be traced, the linkability is useful, as well as *traceable signatures* in [13]. However, the linkability is undesirable in most cases. In case a member leaves voluntarily, the anonymity of signatures before leaving should be ensured. This is the same in case a member's secret key is stolen.

In this paper, we propose VLR group signature schemes from bilinear maps, which moreover satisfy the backward unlinkability. In the schemes, the concept of time intervals is adopted, as [15]. For each member, there are revocation tokens of all intervals $1, \ldots, T$. When a revocation happens at interval j^*, the revocation tokens of the member at all $j \geq j^*$ are sent to verifiers. Then, signatures after j^* (including j^*) can be detected, while signatures before j^* remain anonymous. Therefore, the backward unlinkability holds. Since the proposed schemes adopt only efficient zero-knowledge proofs, signing process is efficient. Moreover, a signer does not need any computation depending on T.

We first propose a basic VLR group signatures scheme, and prove the security. In the basic scheme, a group manager publishes revocation tokens at every interval. Thus, the total data of the revocation tokens published up to a proceeded interval becomes very long. Therefore, we propose an extended scheme, where the total data size is reduced at the sacrifice of the signer's slight cost.

2 Model and Security Definitions

We show a model of VLR group signature scheme with backward unlinkability, which is extended from a model of VLR group signature scheme proposed in [7].

Definition 1. *A VLR group signature scheme with backward unlinkability consists of the following algorithms:*

KeyGen(n, T)**:** *It is a probabilistic algorithm on inputs n, which is the number of members, and T, which is the number of time intervals. It outputs a group public key gpk, an n-element vector of members' secret keys* $\boldsymbol{gsk} =$

$(\boldsymbol{gsk}[1], \ldots, \boldsymbol{gsk}[n])$, and an $n \times T$-element vector of revocation tokens $\boldsymbol{grt} = (\boldsymbol{grt}[1][1], \ldots, \boldsymbol{grt}[n][T])$, where $\boldsymbol{grt}[i][j]$ indicates the token of member i at time interval j.

Sign$(gpk, j, \boldsymbol{gsk}[i], M)$**:** *This takes as inputs the group public key gpk, the current time interval j, a secret key $\boldsymbol{gsk}[i]$, and a message $M \in \{0,1\}^*$, and outputs the signature σ.*

Verify$(gpk, j, RL_j, \sigma, M)$**:** *This takes as inputs gpk, j, a set of the revocation tokens RL_j at the time interval j, a signature σ, and the message M. Then, it outputs either valid or invalid. The validity means that σ is a correct signature on M at interval j w.r.t. gpk, and that the signer is not revoked at the interval j.*

Remark 1. In practice, algorithm **KeyGen** is performed by a trusted group manager. The manager gives each $\boldsymbol{gsk}[i]$ to each group member indexed by i, who can compute a group signature using algorithm **Sign**. Furthermore, at each interval j, the manager distributes revocation list RL_j, which consists of tokens $\boldsymbol{grt}[i][j]$ for all revoked members at j, to verifiers. The verifiers can verify a group signature using algorithm **Verify**.

Then, the security requirements, *Correctness, Traceability*, and *BU-anonymity*, are defined as follows, which are also extended from [7].

Definition 2 (Correctness). *Correctness requires that for all $(gpk, \boldsymbol{gsk}, \boldsymbol{grt}) =$ **KeyGen**(n, T), all $j \in [1, T]$, all RL_j, all $i \in [1, n]$, and all $M \in \{0,1\}^*$,*

$$\mathbf{Verify}(gpk, j, RL_j, \mathbf{Sign}(gpk, j, \boldsymbol{gsk}[i], M), M) = \text{valid} \iff \boldsymbol{grt}[i][j] \notin RL_j.$$

As well as [7], we introduce implicit tracing algorithm: For any interval j, using the revocation token $\boldsymbol{grt}[i][j]$ of all members, the implicit tracing algorithm can trace the signer from a valid signature-message pair (σ, M).

The following traceability requirement captures the unforgeability of group signatures, introduced first by [3]. Consider the following traceability game between an adversary $\mathcal{A}$ and a challenger, where $\mathcal{A}$ tries to forge a signature that cannot be traced to one of members corrupted by $\mathcal{A}$.

Setup: The challenger runs **KeyGen**(n, T), and obtains $gpk, \boldsymbol{gsk}$, and $\boldsymbol{grt}$. He provides $\mathcal{A}$ with gpk and $\boldsymbol{grt}$, and sets U with empty.

Queries: $\mathcal{A}$ can query the challenger about the following.

Signing: $\mathcal{A}$ requests a signature on an arbitrary message M for an arbitrary member i at an arbitrary interval j. The challenger responds the corresponding signature.

Corruption: $\mathcal{A}$ requests the secret key of an arbitrary member i. The challenger adds i to U, and responds the key.

Output: Finally, $\mathcal{A}$ outputs a message M^*, an interval j^*, a set $RL^*_{j^*}$ of revocation tokens, and a signature σ^*.

Then, $\mathcal{A}$ wins if (1) **Verify**$(gpk, j^*, RL^*_{j^*}, \sigma^*, M^*) =$ valid, and (2) σ^* traces to a member outside of the coalition, i.e, $U \setminus RL^*_{j^*}$ or the trace is failure, and (3) $\mathcal{A}$ did not obtain σ^* by making a signing query at M^*.

Definition 3 (Traceability). *Traceability requires that for all PPT $\mathcal{A}$, the probability that $\mathcal{A}$ wins the traceability game is negligible.*

The following BU-anonymity requirement captures the anonymity with the backward unlinkability. Consider the following BU-anonymity game.

Setup: The challenger runs **KeyGen**(n, T), and obtains $gpk, \boldsymbol{gsk}$, and $\boldsymbol{grt}$. He provides $\mathcal{A}$ with gpk.

Queries: At the beginning of every interval $j \in [1, T]$, the challenger announces the beginning of j to $\mathcal{A}$, where j is incremented. At the current interval j, $\mathcal{A}$ can query the challenger about the following.

Signing: $\mathcal{A}$ requests a signature on an arbitrary message M for an arbitrary member i at the current interval j. The challenger responds the corresponding signature.

Corruption: $\mathcal{A}$ requests the secret key of an arbitrary member i.

Revocation: $\mathcal{A}$ requests the revocation of an arbitrary member i at the current interval j. The challenger responds $\boldsymbol{grt}[i][j]$.

Challenge: $\mathcal{A}$ outputs a message M and two members i_0 and i_1. The corruption of i_0 and i_1 must not be requested. Furthermore, the revocations of i_0 and i_1 must not be requested before the current interval j_0 (including j_0). The challenger chooses $\phi \in_R \{0, 1\}$, and responds the signature on M of member i_ϕ at the current interval j_0.

Restricted queries: Similarly, $\mathcal{A}$ can make the signing queries, corruption queries, and revocation queries, while the time interval is incremented. However, $\mathcal{A}$ cannot query the corruptions of i_0 and i_1, and the revocations of i_0 and i_1 at the interval j_0 (Note that the revocations of i_0 and i_1 after j_0 is permitted).

Output: Finally, $\mathcal{A}$ outputs a bit ϕ' indicating its guess of ϕ.

If $\phi' = \phi$, $\mathcal{A}$ wins. We define the advantage of $\mathcal{A}$ as $|\Pr[\phi' = \phi] - 1/2|$.

Definition 4 (BU-anonymity). *BU-anonymity requires that for all PPT $\mathcal{A}$, the advantage of $\mathcal{A}$ on the BU-anonymity game is negligible.*

3 Preliminaries

3.1 Bilinear Groups

Our scheme utilizes bilinear groups and bilinear maps as follows:

1. G_1, G_2 and G' are multiplicative cyclic groups of prime order p,
2. g_1 is a generator of G_1, and g_2 is a generator of G_2,
3. ψ is an efficiently computed isomorphism from G_2 to G_1, with $\psi(g_2) = g_1$,
4. e is an efficiently computed bilinear map: $G_1 \times G_2 \rightarrow G'$, i.e., (1) for all $u \in G_1, v \in G_2$ and $a, b \in Z$, $e(u^a, v^b) = e(u, v)^{ab}$, and (2) $e(g_1, g_2) \neq 1$.

Hereafter, for simplicity, we consider only the case of $G_1 = G_2$, and we set $G = G_1 = G_2$, and $g = g_1 = g_2$. Our scheme can be extended to the case of $G_1 \neq G_2$, as [7].

3.2 Assumptions

Our scheme is based on the q-SDH assumption [6,7] and the DBDH assumption [4] in G.

Definition 5 (q-SDH assumption). *For all PPT algorithm $\mathcal{A}$, the probability*

$$\Pr[\mathcal{A}(g, g^{\gamma}, \ldots, g^{(\gamma^q)}) = (g^{(1/\gamma+x)}, x) \wedge x \in Z_p^*]$$

is negligible, where $\gamma \in_R Z_p^*$.

Definition 6 (Decision BDH (DBDH) assumption). *For all PPT algorithm $\mathcal{A}$, the probability*

$$|\Pr[\mathcal{A}(g, g^a, g^b, g^c, e(g,g)^{abc}) = 0] - \Pr[\mathcal{A}(g, g^a, g^b, g^c, e(g,g)^d) = 0|$$

is negligible, where $a, b, c, d \in_R Z_p^*$.

3.3 Proving Relations on Representations

As well as [6,7], we adopt signatures converted by Fiat-Shamir heuristic (using a hash function) from zero-knowledge proofs of knowledge (PK), where a signer can convince a verifier of knowledge with relations on representations. We call the signatures SPKs. The SPKs we adopt are the generalization of the Schnorr signature, and the underlying PKs are basically derived from [11,12,8]. We introduce the following notation.

$$SPK\{(x_1, \ldots, x_t) : R(x_1, \ldots, x_t)\}(M),$$

which means a signature of message M by a signer who knows secret values $x_1, \ldots, x_t$ satisfying a relation $R(x_1, \ldots, x_t)$. In this paper, the following SPKs on G, G' are utilized.

SPK **of representation:** An SPK proving the knowledge of a representation of $C \in G$ to the bases $g_1, g_2, \ldots, g_t \in G$ on message M is denoted as

$$SPK\{(x_1, \ldots, x_t) : C = g_1^{x_1} \cdots g_t^{x_t}\}(M).$$

This can be also constructed on group G'.

SPK **of representations with equal parts:** An SPK proving the knowledge of representations of $C, C' \in G$ to the bases $g_1, \ldots, g_t \in G$ on message M, where the representations include equal values as parts, is denoted as

$$SPK\{(x_1, \ldots, x_u) : C = g_{i_1}^{x_{j_1}} \cdots g_{i_v}^{x_{j_v}} \wedge C' = g_{i'_1}^{x_{j'_1}} \cdots g_{i'_{v'}}^{x_{j'_{v'}}}\}(M),$$

where indices $i_1, \ldots i_v, i'_1, \ldots i'_{v'} \in \{1, \ldots, t\}$ refer to the bases $g_1, \ldots, g_t$, and indices $j_1, \ldots j_v, j'_1, \ldots, j'_{v'} \in \{1, \ldots, u\}$ refer to the secrets $x_1, \ldots, x_u$. This SPK can be extended for different groups G and G' with the same order p, as follows.

$$SPK\{(x_1, \ldots, x_u) : C = g_{i_1}^{x_{j_1}} \cdots g_{i_v}^{x_{j_v}} \wedge C' = {g'}_{i'_1}^{x_{j_1}} \cdots {g'}_{i'_{v'}}^{x_{j'_{v'}}}\}(M),$$

where $C, g_1, \ldots, g_t \in G$, and $C', g'_1, \ldots, g'_t \in G'$.

In the random oracle model, the SPK can be simulated without the knowledge using a simulator in the zero-knowledge-ness of the underlying PK. Moreover, the SPK has an extractor of the proved secret knowledge given two accepting protocol views whose commitments are the same and whose challenges are different.

4 Proposed Scheme

4.1 Idea

The scheme of [7] is intuitively as follows. For group public key $gpk = (g, g^{\gamma})$, an SDH pair $(A_i = g^{1/(\gamma + x_i)}, x_i)$ is secret key $\boldsymbol{gsk}[i]$ of member i, which is unforgeable without γ. Then, group signature of member i consists of $T_1 = u^r A_i$ and $T_2 = v^r$, where $u, v \in_R G$ and $r \in_R Z_p^*$, and the SPK proving the correctness. The revocation token of member i is A_i. By checking $e(T_1/A, v) = e(u, T_2)$ for all revocation tokens A, it can be checked whether T_1 includes a token of a revoked member.

The proposed scheme is an extension of [7]. To the public key, we add $h_j \in G$ for all $1 \leq j \leq T$, and the secret key is the same. Then, the group signature is modified into $T_3 = e(g^{x_i}, h_j)^r$, $T_4 = g^r$ and the SPK proving the correctness and the ownership of A_i corresponding x_i. The revocation token at interval j is $B_{ij} = h_j^{x_i}$. Then, by checking $T_3 = e(T_4, B)$ for all revocation tokens B at interval j, it can be checked whether T_3 includes a token of a revoked member. On the other hand, the revocation tokens at different interval j' do not satisfy the above checking. Moreover, the computation from a token $h_j^{x_i}$ at j to another $h_{j'}^{x_i}$ at j' is infeasible. Therefore, backward unlinkability is achieved.

4.2 Proposed Algorithms

KeyGen(n, T): This key generation algorithm is given the number of members and the number of time intervals, and computes keys as follows.

1. Select a generator $g \in G$ and $\tilde{g} \in_R G$. Additionally, select $h_j \in_R G$ for all $j \in [1, T]$.
2. Select $\gamma \in_R Z_p^*$ and compute $w = g^{\gamma}$.
3. Select $x_i \in_R Z_p^*$ and compute $A_i = g^{1/(\gamma + x_i)}$ for all $i \in [1, n]$.
4. Compute $B_{ij} = h_j^{x_i}$ for all i and j.

The group public key gpk is $(g, \tilde{g}, h_1, \dots h_T, w)$. Each member's secret key $\boldsymbol{gsk}[i]$ is (A_i, x_i). The revocation token at interval j of member with secret (A_i, x_i) is $\boldsymbol{grt}[i][j] = B_{ij}$. Output $(gpk, \boldsymbol{gsk}, \boldsymbol{grt})$.

Sign$(gpk, j, \boldsymbol{gsk}[i], M)$: The inputs of this signing algorithm are $gpk = (g, \tilde{g}, h_1, \dots h_T, w)$, the current time interval j, the signer's secret $\boldsymbol{gsk}[i] = (A_i, x_i)$ and a signed message $M \in \{0, 1\}^*$. Hereafter, we assume that M includes the time interval j in order to bind the signature to the interval. The algorithm is as follows:

1. Select randoms $\alpha, \beta, \delta \in_R Z_p^*$.
2. Compute $T_1 = A_i \tilde{g}^\alpha$, $T_2 = g^\alpha \tilde{g}^\beta$, $T_3 = e(g^{x_i}, h_j)^\delta$, and $T_4 = g^\delta$.
3. Compute

$$\begin{aligned} V = SPK\{(\alpha, \beta, \delta, x_i, A_i) : T_1 = A_i \tilde{g}^\alpha \wedge T_2 = g^\alpha \tilde{g}^\beta \\ \wedge T_3 = e(g^{x_i}, h_j)^\delta \wedge T_4 = g^\delta \\ \wedge e(A_i, wg^{x_i}) = e(g, g)\}(M). \end{aligned}$$

The detail of this SPK is shown in Section 4.3.

Output the group signature $\sigma = (T_1, T_2, T_3, T_4, V)$.

Verify$(gpk, j, RL_j, \sigma, M)$: The inputs are $gpk = (g, \tilde{g}, h_1, \dots h_T, w)$, the current time interval j, the revocation list RL_j that consists of $\boldsymbol{grt}[i][j]$ for all revoked i at the interval j, a target signature $\sigma = (T_1, T_2, T_3, T_4, V)$ and the message $M \in \{0, 1\}^*$.

1. **Signature check.** Check that σ is valid, by checking the SPK V.
2. **Revocation check.** Check that the signer is not revoked at the interval j, by checking $T_3 \neq e(T_4, B_{ij})$ for all $B_{ij} \in RL_j$.

4.3 Detail of the *SPK*

The SPK V in the algorithm **Sign** is computed as the following SPK V'.

$$\begin{aligned} V' = SPK\{(\alpha, \beta, \delta, x_i, \epsilon, \zeta, \eta) : T_2 = g^\alpha \tilde{g}^\beta \wedge 1 = T_2^{x_i} (1/g)^\epsilon (1/\tilde{g})^\zeta \\ \wedge e(T_1, w)(1/e(g, g)) = (1/e(T_1, g))^{x_i} e(\tilde{g}, w)^\alpha e(\tilde{g}, g)^\epsilon \\ \wedge T_3 = e(g, h_j)^\eta \wedge T_4 = g^\delta \wedge 1 = T_4^{x_i} (1/g)^\eta\}(M). \end{aligned}$$

This SPK can be computed by the SPK for the representations, where $\epsilon = x_i \alpha$, $\zeta = x_i \beta$, and $\eta = x_i \delta$ are adopted.

What we require is to prove that the SPK V is equivalent to V'. The following lemma ensures the equivalence.

Lemma 1. *V' is an SPK of knowledge $(\alpha, \beta, \delta, x_i, A_i)$ s.t.*

$$T_1 = A_i \tilde{g}^\alpha \wedge T_2 = g^\alpha \tilde{g}^\beta \wedge T_3 = e(g^{x_i}, h_j)^\delta \wedge T_4 = g^\delta \wedge e(A_i, wg^{x_i}) = e(g, g).$$

Proof. Since V' is an SPK of knowledge $(\alpha, \beta, \delta, x_i, \epsilon, \zeta, \eta)$ s.t.

$$T_2 = g^\alpha \tilde{g}^\beta \tag{1}$$
$$1 = T_2^{x_i} (1/g)^\epsilon (1/\tilde{g})^\zeta \tag{2}$$
$$e(T_1, w)(1/e(g, g)) = (1/e(T_1, g))^{x_i} e(\tilde{g}, w)^\alpha e(\tilde{g}, g)^\epsilon \tag{3}$$
$$T_3 = e(g, h_j)^\eta \tag{4}$$
$$T_4 = g^\delta \tag{5}$$
$$1 = T_4^{x_i} (1/g)^\eta, \tag{6}$$

such knowledge can be extracted. From the equations (1), (2), the equation $g^{\epsilon}\tilde{g}^{\zeta} = g^{x_i\alpha}\tilde{g}^{x_i\beta}$ holds, and thus $\epsilon = x_i\alpha$ holds. Then, consider the following equation transformed from (3).

$$e(T_1, w)e(T_1, g)^{x_i}/e(\tilde{g}, w)^{\alpha}e(\tilde{g}, g)^{x_i\alpha} = e(g, g).$$

Then, from

$$e(T_1, w)e(T_1, g)^{x_i} = e(T_1, wg^{x_i}) \text{ and } e(\tilde{g}, w)^{\alpha}e(\tilde{g}, g)^{x_i\alpha} = e(\tilde{g}^{\alpha}, wg^{x_i}),$$

we obtain $e(T_1/\tilde{g}^{\alpha}, wg^{x_i}) = e(g, g)$. Thus, setting $A_i = T_1/\tilde{g}^{\alpha}$, we can extract knowledge x_i, A_i s.t.

$$T_1 = A_i\tilde{g}^{\alpha} \text{ and } e(A_i, wg^{x_i}) = e(g, g).$$

On the other hand, from equations (5), (6), we obtain $g^{\eta} = g^{x_i\delta}$. Therefore, from equation (4), we can extract knowledge x_i, δ s.t. $T_3 = e(g^{x_i}, h_j)^{\delta}$. □

4.4 Details of Sign and Verify Algorithms

For efficiency consideration, this subsection describes the **Sign** and **Verify** algorithms of the proposed scheme in Section 4.2, where the *SPK*s for representations shown in Section 4.3 are described in details. The construction of each *SPK* for a representation is similar to that in [7] or Schnorr based *SPK*s on groups with known orders. Thus, we omit the proof that underlying *PK*s of following *SPK*s are zero-knowledge proofs of knowledge.

Sign$(gpk, j, \boldsymbol{gsk}[i], M)$:

1. Select randoms $\alpha, \beta, \delta \in_R Z_p^*$, and set $\epsilon = x_i\alpha$, $\zeta = x_i\beta$, and $\eta = x_i\delta$.
2. Compute $T_1 = A_i\tilde{g}^{\alpha}$, $T_2 = g^{\alpha}\tilde{g}^{\beta}$, $T_3 = e(g^{x_i}, h_j)^{\delta}$, and $T_4 = g^{\delta}$.
3. Compute $SPK\ V'$ (i.e., V) as follows.
   (a) Pick blinding factors $r_\alpha, r_\beta, r_\delta, r_{x_i}, r_\epsilon, r_\zeta, r_\eta \in_R Z_p$.
   (b) Compute

$$\begin{aligned} R_1 &= g^{r_\alpha}\tilde{g}^{r_\beta}, \\ R_2 &= T_2^{r_{x_i}}(1/g)^{r_\epsilon}(1/\tilde{g})^{r_\zeta}, \\ R_3 &= (1/e(T_1, g))^{r_{x_i}}e(\tilde{g}, w)^{r_\alpha}e(\tilde{g}, g)^{r_\epsilon}, \\ R_4 &= e(g, h_j)^{r_\eta}, \\ R_5 &= g^{r_\delta}, \\ R_6 &= T_4^{r_{x_i}}(1/g)^{r_\eta}. \end{aligned}$$

   (c) Compute a challenge $c \in Z_p$ using a hash function H that is regarded as a random oracle.

$$c = H(gpk, j, M, T_1, T_2, T_3, T_4, R_1, R_2, R_3, R_4, R_5, R_6).$$

   (d) Compute $s_\alpha = r_\alpha + c\alpha$, $s_\beta = r_\beta + c\beta$, $s_\delta = r_\delta + c\delta$, $s_{x_i} = r_{x_i} + cx_i$, $s_\epsilon = r_\epsilon + c\epsilon$, $s_\zeta = r_\zeta + c\zeta$, and $s_\eta = r_\eta + c\eta$ in Z_p.

Output the group signature $\sigma = (T_1, T_2, T_3, T_4, c, s_\alpha, s_\beta, s_\delta, s_{x_i}, s_\epsilon, s_\zeta, s_\eta)$.

Verify$(gpk, j, RL_j, \sigma, M)$:

1. **Signature check.** Check that σ is valid, by checking the $SPK\ V'$, as follows.
   (a) Retrieve

$$\begin{aligned}
\tilde{R}_1 &= g^{s_\alpha}\tilde{g}^{s_\beta}(1/T_2)^c,\\
\tilde{R}_2 &= T_2^{s_{x_i}}(1/g)^{s_\epsilon}(1/\tilde{g})^{s_\zeta},\\
\tilde{R}_3 &= (1/e(T_1,g))^{s_{x_i}}e(\tilde{g},w)^{s_\alpha}e(\tilde{g},g)^{s_\epsilon}((1/e(T_1,w))e(g,g))^c,\\
\tilde{R}_4 &= e(g,h_j)^{s_\eta}(1/T_3)^c,\\
\tilde{R}_5 &= g^{s_\delta}(1/T_4)^c,\\
\tilde{R}_6 &= T_4^{s_{x_i}}(1/g)^{s_\eta}.
\end{aligned}$$

   (b) Check the challenge c:

$$c = H(gpk, j, M, T_1, T_2, T_3, T_4, \tilde{R}_1, \tilde{R}_2, \tilde{R}_3, \tilde{R}_4, \tilde{R}_5, \tilde{R}_6).$$

2. **Revocation check.** Check that the signer is not revoked at the interval j, by checking $T_3 \neq e(T_4, B_{ij})$ for all $B_{ij} \in RL_j$.

Signature Length. This group signature includes 3 elements from G, 1 element from G' and 8 elements from Z_p. When an elliptic curve is used as well as [7], p is 170 bits, elements of G are 171 bits, and elements of G' is 1020 bits. In that case, this group signature is 2893 bits or 362 bytes.
Performance. The signature generation requires 10 multi-exponentiations and 1 bilinear map computation (plus 3 bilinear map computations that can be pre-computed). The verification requires 6 multi-exponentiations and $2 + |RL_j|$ bilinear map computations (plus 4 bilinear map computations that can be pre-computed).

5 Security

Since the correctness is straightforward, only BU-anonymity and traceability are shown.

5.1 BU-Anonymity

Theorem 1. *The proposed scheme satisfies the BU-anonymity in the random oracle model under the DBDH assumption.*

The following lemma implies the above theorem.

Lemma 2. *Suppose adversary $\mathcal{A}$ breaks the BU-anonymity of the proposed scheme with the advantage ϵ and q_H hash queries and q_S signature queries. Then, we can construct $\mathcal{B}$ that breaks the DBDH assumption with the advantage $(1/nT - q_S q_H/p)\epsilon$.*

Proof. The input of $\mathcal{B}$ is $(g, g_1 = g^a, g_2 = g^b, g_3 = g^c, Z)$, where $a, b, c \in_R Z_p^*$ and either $Z = e(g, g)^{abc}$ or $Z = e(g, g)^d$ for $d \in_R Z_p^*$. $\mathcal{B}$ decides which Z it is given by communicating with $\mathcal{A}$, as follows.

Setup. $\mathcal{B}$ simulates **KeyGen**(n, T) as follows.

1. $\mathcal{B}$ picks $i^* \in_R [1, n]$ and $j^* \in_R [1, T]$.
   Furthermore, $\mathcal{B}$ selects $\tilde{g} \in_R G$. Additionally, $\mathcal{B}$ selects $r_j \in_R Z_p^*$ and computes $h_j = g^{r_j}$ for all $j \in [1, T]$ except j^*. For j^*, $\mathcal{B}$ sets $h_{j^*} = g_2 = g^b$.
2. As usual, $\mathcal{B}$ selects $\gamma \in_R Z_p^*$ and computes $w = g^\gamma$.
3. As usual, $\mathcal{B}$ selects $x_i \in_R Z_p^*$ and computes $A_i = g^{1/(\gamma+x_i)}$ for all $i \in [1, n]$ except i^*. For i^*, define $x_{i^*} = a$ and $A_{i^*} = g^{1/(\gamma+a)}$, which are unknown for $\mathcal{B}$.
4. As usual, $\mathcal{B}$ computes $B_{ij} = h_j^{x_i}$ for all i except i^* and all j. For i^*, $\mathcal{B}$ sets $B_{i^*j} = g_1^{r_j} = g^{ar_j} = h_j^a$ except for j^*. For i^* and j^*, define $B_{i^*j^*} = g^{ab} = h_{j^*}^{x_{i^*}}$, which is also unknown.

Note that simulated h_j and B_{ij} have the same distributions as the real, since $a, b, x_i, r_j \in_R Z_p^*$.

Hash queries. At any time, $\mathcal{A}$ can query the hash function used in SPK. $\mathcal{B}$ responds with random values with consistency.

Phase 1. $\mathcal{A}$ can request signing queries, corruption queries, and revocation queries at any time interval j. If $i \neq i^*$, then $\mathcal{B}$ uses the secret key of i to respond to the query as usual. If $i = i^*$, $\mathcal{B}$ responds as follows.

Signing queries: $\mathcal{B}$ computes a simulated group signature of i^*, as follows.

1. $\mathcal{B}$ selects $\delta \in_R Z_p^*$.
2. $\mathcal{B}$ selects $T_1, T_2 \in_R G$. Furthermore, $\mathcal{B}$ computes $T_3 = e(g_1, h_j)^\delta = e(g^a, h_j)^\delta = e(g^{x_{i^*}}, h_j)^\delta$, and $T_4 = g^\delta$.
3. $\mathcal{B}$ computes the simulated SPK V by using the simulator of the perfect zero-knowledge-ness, which includes the backpatch of the hash function. If the backpatch is failure, $\mathcal{B}$ outputs a random guess $\omega' \in_R \{0, 1\}$ and aborts.

Then, $\mathcal{B}$ responds signature $\sigma = (T_1, T_2, T_3, T_4, V)$ to $\mathcal{A}$. Note that each value in σ has the same distribution as the real, since $\alpha, \beta \in_R Z_p^*$ in the real and $T_1, T_2 \in_R G$ in the simulation, and due to the perfect zero-knowledge-ness of SPK.

Revocation queries: If $j \neq j^*$, $\mathcal{B}$ responds B_{i^*j}. Otherwise (i.e., $j = j^*$), $\mathcal{B}$ outputs a random guess $\omega' \in_R \{0, 1\}$ and aborts.

Corruption queries: $\mathcal{B}$ outputs a random guess $\omega' \in_R \{0, 1\}$ and aborts.

Challenge. $\mathcal{A}$ outputs a message M, the current time interval j and two members i_0, i_1 to be challenged. If $j \neq j^*$, $\mathcal{B}$ outputs a random guess $\omega' \in_R \{0, 1\}$ and aborts. Otherwise, $\mathcal{B}$ picks $\phi \in_R \{0, 1\}$. Then, if $i_\phi \neq i^*$, $\mathcal{B}$ outputs a random guess $\omega' \in_R \{0, 1\}$ and aborts. Otherwise, $\mathcal{B}$ responds the following simulated group signature of i^* and j^*.

1. $\mathcal{B}$ regards c as δ, which is unknown.
2. $\mathcal{B}$ selects $T_1, T_2 \in_R G$. Furthermore, $\mathcal{B}$ sets $T_3 = Z$ and $T_4 = g_3 = g^c$. Note that if $Z = e(g,g)^{abc}$, $T_3 = e(g^a, g^b)^c = e(g^{x_{i^*}}, h_{j^*})^{\delta}$.
3. $\mathcal{B}$ computes the simulated $SPK\ V$ by using the simulator of the perfect zero-knowledge-ness.

Phase 2. This is the same as Phase 1.

Output. $\mathcal{A}$ outputs its guess $\phi' \in \{0,1\}$. If $\phi = \phi'$, $\mathcal{B}$ outputs $\omega' = 1$ (implying $Z = (g,g)^{abc}$), and otherwise outputs $\omega' = 0$ (implying $Z = (g,g)^d$).

Now, we evaluate the advantage of the guess of $\mathcal{B}$. Let $\omega \in \{0,1\}$ denote whether the input Z is $e(g,g)^d$ ($\omega = 0$) or $e(g,g)^{abc}$ ($\omega = 1$). Let **abort** be the event that $\mathcal{B}$ aborts. Then, we have $\Pr[\omega = \omega' | \textbf{abort}] = 1/2$. On the other hand, assume that $\mathcal{B}$ does not abort. If $\omega = 0$, i.e., $Z = e(g,g)^d$, then the challenged signature has no information on x_{i^*}. Thus, $\Pr[\omega' = 0 | \overline{\textbf{abort}} \wedge \omega = 0] = 1/2$. If $\omega = 1$, i.e., $Z = e(g,g)^{abc}$, then $\mathcal{B}$ perfectly simulates the real and thus $\mathcal{A}$ guesses correctly with the advantage ϵ. Therefore, we obtain $\Pr[\omega' = 1 | \overline{\textbf{abort}} \wedge \omega = 1] = 1/2 + \epsilon$.

Putting everything together, we obtain the advantage of $\mathcal{B}$'s guess, as follows.

$$\begin{aligned}
&|\Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^{abc}) = 0] - \Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^d) = 0|\\
&= |\Pr[\omega' = 0 | \omega = 1] - \Pr[\omega' = 0 | \omega = 0]|\\
&= |(1 - \Pr[\omega' = 1 | \omega = 1]) - \Pr[\omega' = 0 | \omega = 0]|\\
&= |1 - \Pr[\textbf{abort}]\Pr[\omega' = 1 | \textbf{abort} \wedge \omega = 1]\\
&\qquad - \Pr[\overline{\textbf{abort}}]\Pr[\omega' = 1 | \overline{\textbf{abort}} \wedge \omega = 1]\\
&\qquad - \Pr[\textbf{abort}]\Pr[\omega' = 0 | \textbf{abort} \wedge \omega = 0]\\
&\qquad - \Pr[\overline{\textbf{abort}}]\Pr[\omega' = 0 | \overline{\textbf{abort}} \wedge \omega = 0]|\\
&= |1 - \Pr[\textbf{abort}](\frac{1}{2} + \frac{1}{2}) - \Pr[\overline{\textbf{abort}}]((\frac{1}{2} + \epsilon) + \frac{1}{2})|\\
&= \Pr[\overline{\textbf{abort}}]\epsilon.
\end{aligned}$$

In the rest, we evaluate $\Pr[\overline{\textbf{abort}}]$. If the guesses of i^* and j^* are correct, $\mathcal{B}$ aborts only when the backpatch is failure in the signing query. The probability that a specific signature causes the failure is at most q_H/p, as well as [7]. Thus, for all signature queries, the probability that $\mathcal{B}$ aborts due to the failure of the backpatch is at most $q_S q_H/p$. On the other hand, since $\mathcal{A}$ has no information on i^* and j^* and $\phi \in_R \{0,1\}$, the probability that $\mathcal{B}$ correctly guesses i^* and j^* is at least $1/nT$. Thus, $\Pr[\overline{\textbf{abort}}] \geq 1/nT - q_S q_H/p$.

Therefore, the advantage that $\mathcal{B}$' guesses ω is at least $(1/nT - q_S q_H/p)\epsilon$. □

5.2 Traceability

Theorem 2. *The proposed scheme satisfies the traceability in the random oracle model under the SDH assumption.*

The following lemma implies the above theorem.

Lemma 3. *Suppose adversary $\mathcal{A}$ breaks the traceability of the proposed scheme with the advantage ϵ and q_H hash queries and q_S signature queries. Then, we can construct $\mathcal{B}$ that breaks the $(n+1)$-SDH assumption with the advantage $(\epsilon/n - 1/p)/(16q_H)$.*

Proof sketch. This is similar to the proof in [7]. Consider the following framework with $\mathcal{A}$.

Setup. It is given g, $w = g^{\gamma}$, and n pairs (A_i, x_i). For each $i \in [1, n]$, either $s_i = 1$ indicating that an SDH pair (A_i, x_i) is known, or $s_i = 0$ indicating that x_i is known but A_i is unknown. Furthermore, as usual, choose $\tilde{g}, h_j \in_R G$ for all $j \in [1, T]$ and compute $B_{ij} = h_j^{x_i}$ for all i, j. Then, run $\mathcal{A}$ on $gpk = (g, \tilde{g}, h_1, \ldots, h_T, w)$ and $\boldsymbol{grt} = (B_{11}, \ldots, B_{nT})$.

Hash queries. At any time, $\mathcal{A}$ can query the hash function used in SPK. Respond with random values with consistency.

Signing queries. $\mathcal{A}$ queries a signature on message M at member i and interval j. If $s_i = 1$, respond a signature using the secret key (A_i, x_i). If $s_i = 0$, pick $T_1, T_2 \in_R G$ and $\delta \in_R Z_p^*$ and compute $T_3 = e(g^{x_i}, h_j)^{\delta}$ and $T_4 = g^{\delta}$. Furthermore, obtain a simulated SPK V using the simulator of the SPK, which includes the backpatch of the hash function. Respond (T_1, T_2, T_3, T_4, V).

Corruption queries. $\mathcal{A}$ requests the secret key at member i. If $s_i = 0$, then abort. Otherwise, respond requested key (A_i, x_i).

Output. Finally, $\mathcal{A}$ outputs a forged signature $\sigma^* = (T_1^*, T_2^*, T_3^*, T_4^*, V^*)$ including a secret key A^*. Using all B_{ij}, we can identify the member. If the identification fails (i.e., the member is outside of all i), output σ. Otherwise, some i is identified. If $s_i = 0$, then output σ. Otherwise (i.e., $s_i = 1$), abort.

Then, there are two types of forger on the above framework. Type 1 forger forges a signature of the member who is different from all i. Type 2 forger forges a signature of the member i whose corruption is not requested.

For q-SDH instance $(g, g^{\gamma}, \ldots, g^{\gamma^q})$, we can obtain $g, w = g^{\gamma}$ and $q-1$ SDH pairs (A_i, x_i) s.t. $e(A_i, g^{x_i} w) = e(g, g)$, using the technique of [5]. On the other hand, any SDH pair besides these $q-1$ pairs can be transformed a solution of the q-SDH instance, which means that the q-SDH assumption is broken, using the same technique. As well as [7], we treat two types of forger differently.

Type 1. Given $(n+1)$-SDH instance, obtain n SDH pairs (A_i, x_i) with (g, w). Then, perform the framework with Type 1 forger $\mathcal{A}$ (i.e., all $s_i = 1$). $\mathcal{A}$ finally outputs a signature with secret key A^* s.t. $A^* \neq A_i$ for all i. In this case, the simulation is perfect, and thus $\mathcal{A}$ succeeds with advantage ϵ.

Type 2. Given n-SDH instance, obtain $n-1$ SDH pairs (A_i, x_i), which distributes n pairs, and set $s_i = 1$. For the unfilled entry at random index i^*, select $x_{i^*} \in_R Z_p^*$ (A_{i^*} is unknown), and set $s_{i^*} = 0$. Then, perform the framework with type 2 forger $\mathcal{A}$. In this case, it succeeds only if $\mathcal{A}$ never requests the corruption of i^*, but forges the signature including A_{i^*}. As discussed in [7], the value of i^* is independent $\mathcal{A}$'s view. Thus, the probability that $\mathcal{A}$ outputs the signature of i^* is at least ϵ/n.

Now we show how to obtain another SDH pair beyond the given $q-1$ SDH pairs, using the framework with Type 1 or Type2. We can rewind the framework to obtain two forged signatures on the same message M and the same interval j, where the commitments in the $SPK\ V$ are the same but the challenges and responses are different. As shown in [7], by the forking lemma, the successful probability is at least $(\epsilon' - 1/p)^2/(16q_H)$, where ϵ' is the probability that the framework on each forger succeeds. Thus, using the extractor of the $SPK\ V$, we can obtain a pair (A^*, x^*) s.t. $A^* \neq A_i$ and $x^* \neq x_i$ for all i with the probability $(\epsilon' - 1/p)^2/(16q_H)$.

Putting everything together, we have shown the following. Using Type 1 forger, we can solve the $(n+1)$-SDH instance with $(\epsilon - 1/p)^2(16/q_H)$. Using Type 2 forger, we can solve the n-SDH instance with $(\epsilon/n - 1/p)^2(16/q_H)$. We can guess the type of forger with the probability 1/2. Therefore, the pessimistic Type 2 forger proves the theorem. □

6 Extension

In practice, the revocation tokens $B_{ij} \in RL_j$ are published at the beginning of each interval j, where the group manager adds the revocation tokens to a public directory. Verifiers fetch needed RL_j from the directory on demand. Since the revocation can be performed only at the beginning of each interval, parameter T should be large. Furthermore, it is general that the list RL_j becomes longer as interval j proceeds. Therefore, all data in the public directory at a proceeded interval j, i.e., $RL_1, \ldots, RL_j$ becomes very long.

In this section, we propose an extended scheme, where the data size of the published revocation tokens in the public directory is reduced at the sacrifice of the signer's slight cost.

At first, modify $h_1, \ldots, h_T$ as follows. Consider a k-ary tree with two levels for an integer k s.t. $T \leq k^2$ (see Fig. 1). Although we show only the case of two levels, the extension to more levels is easy. In the tree, the root node is N_0, N_{j_1} is the j_1-th child of N_0, and $N_{j_1 j_2}$ is the j_2-th child of N_{j_1}, for $j_1, j_2 \in [1, k]$. Each node N_{j_1} is assigned to $h_{j_1} \in_R G$, and each node $N_{j_1 j_2}$ is assigned to

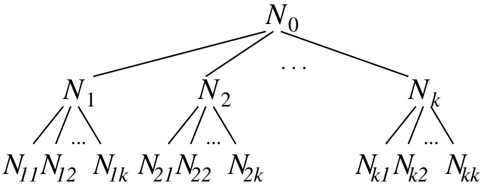

Fig. 1. A k-ary tree with two levels

$h_{j_1 j_2} \in_R G$. In this situation, every interval $j \in [1, T]$ can be correspondent to a pair of two indexes j_1 and j_2 for $j_1, j_2 \in [1, k]$ such that $j = j_1 k + j_2$. Then, the next interval of (j_1, j_2) is $(j_1, j_2 + 1)$ unless $j_2 \neq k$, and if $j_2 = k$, the next interval is $(j_1 + 1, 1)$. In each interval (j_1, j_2), the values h_{j_1} and $h_{j_1 j_2}$ along the path are used.

A group signature on message M by member i at interval (j_1, j_2) is computed as $T_1 = A_i \tilde{g}^{\alpha}$, $T_2 = g^{\alpha} \tilde{g}^{\beta}$, $T_3 = e(g^{x_i}, h_{j_1})^{\delta}$, $T_4 = g^{\delta}$, $T_3' = e(g^{x_i}, h_{j_1 j_2})^{\delta'}$, and $T_4' = g^{\delta'}$ for $\alpha, \beta, \delta, \delta' \in_R Z_p^*$, together with the following SPK V.

$$\begin{aligned} V = SPK\{(\alpha, \beta, \delta, \delta', x_i, A_i) : T_1 = A_i \tilde{g}^{\alpha} \wedge T_2 = g^{\alpha} \tilde{g}^{\beta} \\ \wedge T_3 = e(g^{x_i}, h_{j_1})^{\delta} \wedge T_4 = g^{\delta} \\ \wedge T_3' = e(g^{x_i}, h_{j_1 j_2})^{\delta'} \wedge T_4' = g^{\delta'} \\ \wedge e(A_i, w g^{x_i}) = e(g, g)\}(M). \end{aligned}$$

The difference between the basic scheme and this extended scheme is the parts T_3, T_4, T_3', T_4' and V. On the other hand, revocation token $B_{i(j_1, j_2)}$ for member i at interval (j_1, j_2) is a pair $(B_{ij_1} = h_{j_1}^{x_i}, B_{ij_1 j_2} = h_{j_1 j_2}^{x_i})$. Then, for $B_{ij_1 j_2}$, by checking $T_3' = e(T_4', B_{ij_1 j_2})$, it can be detected whether a group signature was made by member i at interval (j_1, j_2). On the other hand, for B_{ij_1}, by checking $T_3 = e(T_4, B_{ij_1})$, group signatures of i at intervals $(j_1, *)$ can be detected, where $*$ means any value of $[1, k]$. Namely, one level of tokens (the upper level, tokens of the form B_{ij_1}) allows to revoke an user during $k \approx \sqrt{T}$ time intervals at once.

Consider how to publish the revocation tokens as follows. Assume that member i is revoked at interval (j_1^*, j_2^*). Then, if $j_2^* \neq 1$, the manager publishes $B_{ij_1^* j_2^*} = h_{j_1^* j_2^*}^{x_i}$ in the public directory. Afterward, at each interval (j_1^*, j_2) s.t. $j_2^* < j_2 \leq k$, the manager similarly publishes $B_{ij_1^* j_2}$. After that, at every interval $(j_1, 1)$ s.t. $j_1^* < j_1 \leq k$, the manager publishes $B_{ij_1} = h_{j_1}^{x_i}$. Note that the manager does not publish $B_{ij_1 j_2}$ any longer. If $j_2^* = 1$, B_{ij_1} is only published at every interval $(j_1, 1)$ for $j_1^* \leq j_1 \leq k$.

In the extended scheme, the manager has only to publish revocation tokens per $k \approx \sqrt{T}$ intervals, except for the initial overhead (i.e., the publication of $B_{ij_1^* j_2}$). Thus, the total size of revocation tokens in public directory is sufficiently reduced. On the other hand, the signer has to compute T_3', T_4' and the corresponding SPK additionally. The communication overhead is 1531 bits and the computational overhead is 6 exponentiations (plus 1 bilinear map computations that can be pre-computed). The security of the extended scheme can be easily proved in the similar way to the basic one.

7 Concluding Remarks

Based on the bilinear maps, we have proposed a VLR group signature scheme with the backward unlinkability, and extended it to a scheme where the published revocation tokens are reduced.

In the proposed scheme, after a revocation, the revoked member remains excluded forever. However, it is easily extended to the scheme where the member is excluded only for specific intervals. This property is useful in some applications.

An open problem is to construct a shorter VLR group signature scheme with the backward unlinkability. Our group signature includes an elements of G', which is longer than elements of G. It is better to construct a signature from only elements of G.

Acknowledgments

We would like to thank the anonymous reviewers for helpful comments.

References

1. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik, “A practical and provably secure coalition-resistant group signature scheme,” Advances in Cryptology — CRYPTO 2000, LNCS 1880, pp.255–270, Springer–Verlag, 2000.
2. G. Ateniese, D. Song, and G. Tsudik, “Quasi-efficient revocation of group signatures,” Proc. 6th Financial Cryptography Conference (FC 2002), LNCS 2357, pp.183–197, Springer–Verlag, 2003.
3. M. Bellare, D. Micciancio, and B. Warinschi, “Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions,” Advances in Cryptology — EUROCRYPT 2003, LNCS 2656, pp.614–629, Springer–Verlag, 2003.
4. D. Boneh and X. Boyen, “Efficient selective-ID secure identity-based encryption without random oracles,” Advances in Cryptology — EUROCRYPT 2004, LNCS 3027, pp.223–238, Springer–Verlag, 2004.
5. D. Boneh and X. Boyen, “Short signatures without random oracles,” Advances in Cryptology — EUROCRYPT 2004, LNCS 3027, pp.56–73, Springer–Verlag, 2004.
6. D. Boneh, X. Boyen, and H. Shacham, “Short group signatures,” Advances in Cryptology — CRYPTO 2004, LNCS 3152, pp.41–55, Springer–Verlag, 2004.
7. D. Boneh and H. Shacham, “Group signatures with verifier-local revocation,” Proc. 11th ACM Conference on Computer and Communications Security (ACM-CCS ’04), pp.168–177, 2004.
8. J. Camenisch and M. Stadler, “Efficient group signature schemes for large groups,” Advances in Cryptology — CRYPTO ’97, LNCS 1294, pp.410–424, Springer–Verlag, 1997.
9. J. Camenisch and A. Lysyanskaya, “Dynamic accumulators and application to efficient revocation of anonymous credentials,” Advances in Cryptology — CRYPTO 2002, LNCS 2442, pp.61–76, Springer–Verlag, 2002.
10. D. Chaum and E. van Heijst, “Group signatures,” Advances in Cryptology — EUROCRYPT ’91, LNCS 547, pp.241–246, Springer–Verlag, 1991.
11. D. Chaum, J.H. Evertse, and J. van de Graaf, “An improved protocol for demonstrating possession of discrete logarithms and some generalizations,” Advances in Cryptology — EUROCRYPT ’87, LNCS 304, pp.127–141, Springer–Verlag, 1988.
12. D. Chaum and T.P. Pedersen, “Wallet databases with observers,” Advances in Cryptology — CRYPTO ’92, LNCS 740, pp.89–105, Springer–Verlag, 1993.

13. A. Kiayias, Y. Tsiounis, and M. Yung, "Traceable signatures," Advances in Cryptology — EUROCRYPT 2004, LNCS 3027, pp.571–589, Springer–Verlag, 2004.
14. L. Nguyen and R. Safavi-Naini, "A trapdoor-free and efficient group signature scheme from bilinear pairings," Advances in Cryptology — ASIACRYPT 2004, Springer–Verlag, 2004.
15. D.X. Song, "Practical forward secure group signature schemes," Proc. 8th ACM Conference on Computer and Communications Security (ACM-CCS '01), pp.225–234, 2001.
16. G. Tsudik and S. Xu, "Accumulating composites and improved group signing," Advances in Cryptology — ASIACRYPT 2003, LNCS 2894, pp.269–286, Springer–Verlag, 2003.