Revocable Group Signature Schemes with Constant Costs for Signing and Verifying

Toru Nakanishi, Hiroki Fujii, Yuta Hira, 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. Lots of revocable group signature schemes have been proposed so far. In one type of revocable schemes, signing and/or verifying algorithms have $O(N)$ or $O(R)$ complexity, where N is the group size and R is the number of revoked members. On the other hand, in Camenisch-Lysyanskaya scheme and the followers, signing and verifying algorithms have $O(1)$ complexity. However, before signing, updates of the secret key are required. The complexity is $O(R)$ in the worst case. In this paper, we propose a revocable scheme with signing and verifying of $O(1)$ complexity, where no updates of secret key are required. The compensation is the long public key of $O(N)$. In addition, we extend it to the scheme with $O(\sqrt{N})$-size public key, where signing and verifying have constant extra costs.

1 Introduction

Group signatures [15] allow a signer to sign a message anonymously as a group member, while only a designated trusted party can identify the signer from the signature. The group is managed by a group manager (GM) who permits a user to join the group. For simplicity, this paper assumes that GM is also the trusted party to identify the signer (This assumption can be easily relaxed). The applications of group signatures include anonymous credentials, direct anonymous attestations, and ID management reported in [13, 11, 20].

Toward making the group signatures practicable, Boneh et al. have proposed a short group signature scheme based on pairings [7], where signatures are shorter than existing RSA-based group signature schemes. With the advance of the implementations of pairings (e.g., [3, 19]), we can obtain the implementations of the signing/verifying of the group signatures with practical computation times and data sizes, if the revocation is neglected. The revocation means that the membership of a group member can be easily revoked to exclude the member from the group.

Lots of revocable group signature schemes have been proposed so far (e.g, [10, 14, 24, 7, 8, 12]). However, one type of schemes [10, 24, 8] has a disadvantage: Singing and/or verifying have the computational complexity (exponentiations or pairings) of $O(N)$ or $O(R)$, where N is the group size and R is the number of revoked members. Thus, these schemes are not suitable for large and dynamic groups.

Camenisch and Lysyanskaya proposed an elegant revocable scheme [14] using dynamic accumulators. In this scheme, the complexity of signing/verifying is $O(1)$ w.r.t. N and R. However, the disadvantage of this scheme is that, whenever making a signature, the signer has to modify his secret key. The modification requires some data of joining and removed members since the last time he signed. This implies that a signer requires a computation of $O(N)$. This is relaxed into $O(R)$ in [12, 7]. However, since R becomes large in large and dynamic groups, we should explore another approach for signing/verifying with $O(1)$ costs.

Another RSA-based approach without the key update is proposed in [23]. In the approach, the large group is partitioned to sub-groups, and the scheme of [24] is utilized for each smaller sub-groups. The scheme of [24] achieves $O(1)$ exponentiations on signing/verifying for middle-scale sub-groups with less than about 1000 members. The computational cost on partitioning in [23] is also $O(1)$. Thus, this approach achieves signing/verifying with $O(1)$ costs. However, this approach has a weakness for larger groups; The signer has to obtain public data reflecting the current membership situation for each member, but the size of the data is $O(N)$. Namely, the signer has to fetch the data with $O(N)$ size whenever signing. Another problem is that it is based on RSA, and thus the long RSA modulus leads to long signatures and revocation data. Therefore, for larger groups, the communication cost becomes vast.

Our contributions. In this paper, we propose a pairing-based revocable scheme satisfying

1. signing/verifying requires only $O(1)$ computational costs,
2. any update of member's secret key is not required, and
3. data related to revocation, which is fetched by the signer, has $O(R)$ size.

Therefore, signing/verifying is sufficiently fast even for larger or dynamic groups, while the communication does not cause large delay.

On the other hand, the compensation is the long public key with $O(N)$ size. In general applications (e.g., authentications for Web services), the public key is distributed once in joining, together with the software of the applications. Thus, the compensation is not so serious. However, for millions of members, the long time of downloading may disgust users. To solve this, in this paper, we also propose an extended scheme with $O(\sqrt{N})$-size public key, where signing/verifying has constant extra costs.

The reduction to the $O(\sqrt{N})$ size means the sufficient practicality of our schemes for large groups. In case of the 112-bit security level and $N = 1,000,000$, the public key size of the extended scheme is less than 100 KBytes. This size shows the sufficient practicality of the storage, not only in usual PCs but also in smart phones. Furthermore, since clients have only to download the public key once, the communication cost does not matter.

We can consider a trivial revocation method, where a non-revoked member fetches a new secret key whenever revocation happens or a time interval proceeds.

To fetch it anonymously, the signer has to fetch the keys of whole non-revoked members. This means fetching $O(N)$-size data. In our scheme, the fetched data is reduced to $O(R)$ size.

The signer's fetching $O(R)$-size date is a concern for the realization of group signatures. However, except the VLR (Verifier-Local Revocation) method [8], every revocation mechanism such as [10, 14, 7, 12, 24] needs such a communication cost. Note that the VLR mechanism requires $O(R)$ computational cost in the verification. Thus, we consider that our revocation method is currently better than the other methods in the natural situation that the signer can use a broadband channel. The reduction in the fetched data is an open problem.

Related works. One of trends in researches on group signatures is to exclude the random oracle model. From the viewpoint of both efficiency and security, Groth's scheme [18] is currently the best choice. Although this scheme achieves the constant length of signatures, it still is greatly less efficient than efficient schemes [7, 16] in the random oracle model. Since our aim is the realization of group signatures in practical applications, we adopt the efficient underlying scheme [16] to add our novel revocation mechanism. This means that our scheme is secure in the random oracle model. A construction without the random oracle is one of our future works.

2 Model and Security Definitions

We show a model of revocable group signature scheme. This model and the following security requirements are derived from [8, 5]. Mainly, for the simplicity, we construct our model on the basis of the model of [8]. The differences from [8] are the revocation mechanism, **Join** algorithm (and non-frameability), and **Open** algorithm. In [8], **Verify** algorithm is given revocation list $\boldsymbol{RL}_t$, but in our model, **Sign** algorithm is given $\boldsymbol{RL}_t$. We consider that the non-frameability (i.e., preventing GM from forging a member's signature) is important, and thus, add **Join** algorithm and the definition of the non-frameability, based on [5]. Also, the **Open** algorithm is added for the purpose of identifying an illegal member.

Revocable group signature scheme consists of the following algorithms:

Setup: This probabilistic initial setup algorithm, on input 1^ℓ, outputs public parameters $param$.

KeyGen: This probabilistic key generation algorithm for GM, on input N that is the maximum number of members and $param$, outputs the group public key gpk and GM's secret key msk.

Join: This is an interactive protocol between a probabilistic algorithm **Join-U** for the i-th user and a probabilistic algorithm **Join-GM** for GM, where the user joins the group controlled by GM w.r.t. gpk. **Join-U**, on input gpk, outputs $\boldsymbol{usk}[i]$ that is the user's secret key. On the other hand, **Join-GM**, on inputs gpk, msk, outputs $\boldsymbol{reg}[i]$, which means the registration log of the i-th user.

$\boldsymbol{usk}$ denotes the list of all users' secret keys, $\boldsymbol{reg}$ denotes the list of all users' registration log, and i denotes user's ID. We index the secret key and registration log of user i by $\boldsymbol{usk}[i]$ and $\boldsymbol{reg}[i]$, respectively.

Revoke: This probabilistic algorithm, on inputs gpk, t and RU that is a set of revoked members' IDs, outputs revocation list $\boldsymbol{RL}_t$.
t denotes a time counter, and $\boldsymbol{RL}_t$ denotes the revocation list of data on revoked users at time t.

Sign: This probabilistic algorithm, on inputs gpk, $\boldsymbol{usk}[i]$, t, $\boldsymbol{RL}_t$, and signed message M, outputs the signature σ.

Verify: This is a deterministic algorithm for verification. The input is gpk, t, a signature σ, and the message M. Then the output is 'valid' or 'invalid'.

Open: This deterministic algorithm, on inputs gpk, msk, $\boldsymbol{reg}$, t, σ and M, outputs i, which indicates the signer of σ.

The security requirements, *Traceability, Anonymity*, and *Non-frameability*, are defined as follows.

2.1 Traceability

The following traceability requirement captures the unforgeability and the revocability of group signatures. 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 $\mathcal{A}$ or to forge a signature that is traced to a revoked member corrupted by $\mathcal{A}$.

Setup: The challenger runs **Setup** and **KeyGen**, and obtains gpk and msk. He provides $\mathcal{A}$ with gpk, and run $\mathcal{A}$. He sets $t = 0$ and CU and RU with empty, where CU denotes the set of IDs of users corrupted by $\mathcal{A}$, and RU denotes the set of IDs of revoked users.

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

- **H-Join:** $\mathcal{A}$ can request the i-th user's join. Then, the challenger executes the join protocol, where the challenger plays the both role of the joining user and GM.
- **C-Join:** $\mathcal{A}$ can request the i-th user's join. Then, $\mathcal{A}$ as the joining user executes the join protocol with the challenger as GM. The challenger adds i to CU.
- **Revocation:** $\mathcal{A}$ requests the revocation of a member i. The challenger increases t by 1, add i to RU, and responds $\boldsymbol{RL}_t$ for t and RU.
- **Signing:** $\mathcal{A}$ requests a signature on a message M for a member i. The challenger responds the corresponding signature using the current $\boldsymbol{RL}_t$, if $i \notin CU$.
- **Corruption:** $\mathcal{A}$ requests the secret key of a member i. The challenger responds $\boldsymbol{usk}[i]$ if $i \notin CU$. The challenger adds i to CU.
- **Open:** $\mathcal{A}$ requests to open a signature σ on the message M. The challenger responds the corresponding signer's ID i.

Output: Finally, $\mathcal{A}$ outputs a message M^* and a signature σ^* on the current $\boldsymbol{RL}_t$.

Then, $\mathcal{A}$ wins if

1. $\mathbf{Verify}(gpk, t, \sigma^*, M^*) = \text{valid}$,
2. for $i^* = \mathbf{Open}(gpk, msk, \boldsymbol{reg}, t, \sigma^*, M^*)$, $i^* \notin CU \setminus RU$, and
3. $\mathcal{A}$ did not obtain σ^* by making a signing query at M^*.

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

2.2 Anonymity

The following anonymity requirement captures the anonymity and unlinkability of signatures. Consider the following anonymity game.

Setup: The challenger runs **KeyGen**, and obtains gpk and msk. He provides $\mathcal{A}$ with gpk, and run $\mathcal{A}$. He sets $t = 0$ and RU and CU with empty.

Queries: $\mathcal{A}$ can query the challenger. The available queries are the same ones as in the traceability game.

Challenge: $\mathcal{A}$ outputs a message M and two members i_0 and i_1. If $i_0 \notin CU$ and $i_1 \notin CU$, the challenger chooses $\phi \in_R \{0, 1\}$, and responds the signature on M of member i_ϕ using the current $\boldsymbol{RL}_t$.

Restricted queries: Similarly, $\mathcal{A}$ can make the queries. However, $\mathcal{A}$ cannot query opening of the signature responded in the challenge.

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 $|\Pr[\phi' = \phi] - 1/2|$.

Anonymity requires that for all PPT $\mathcal{A}$, the advantage of $\mathcal{A}$ on the anonymity game is negligible.

2.3 Non-frameability

This property requires that a signature of an honest member cannot be computed by other members and even GM.

Consider the following non-frameability game.

Setup: The challenger uses **Setup** to obtain $param$, and sets $t = 0$ and RU and HU with empty, where HU denotes the set of IDs of honest users who are not corrupted by $\mathcal{A}$. Then, run $\mathcal{A}$ on $param$, who initially outputs gpk.

Queries: After the key output in the run, $\mathcal{A}$ issues the following queries to the challenger.

- **Join:** $\mathcal{A}$ can request the i-th honest user's join. Then, $\mathcal{A}$ as GM executes the join protocol with the challenger as the i-the user. The challenger adds i to HU.
- **Revocation:** $\mathcal{A}$ can request the revocation of member i. The challenger increases t by 1, add i to RU, and responds $\boldsymbol{RL}_t$ for t and RU.
- **Sign:** $\mathcal{A}$ can request a signature on message M for user's ID i using the current $\boldsymbol{RL}_t$. The challenger replies $\mathbf{Sign}(gpk, \boldsymbol{usk}[i], t, \boldsymbol{RL}_t, M)$, if $i \in HU$.

Corruption: $\mathcal{A}$ can request to corrupt a member by sending the member's ID i. The challenger returns $\boldsymbol{usk}[i]$, if $i \in HU$. The challenger deletes i from HU.

Output: Finally, $\mathcal{A}$ outputs a message M^* and a signature σ^*.

Then, $\mathcal{A}$ wins if

1. $\mathbf{Verify}(gpk, t, \sigma^*, M^*) =$ valid,
2. for $i^* = \mathbf{Open}(gpk, msk, \boldsymbol{reg}, t, \sigma^*, M^*)$, $i^* \in HU$, and
3. $\mathcal{A}$ did not obtain σ^* by making a signing query at M^*.

Non-Frameability requires that for all PPT $\mathcal{A}$, the probability that $\mathcal{A}$ wins the non-frameability game is negligible.

3 Preliminaries

3.1 Bilinear Groups

Our scheme utilizes bilinear groups and bilinear maps as follows:

1. $\mathcal{G}$, $\mathcal{H}$ and $\mathcal{T}$ are multiplicative cyclic groups of prime order p,
2. g and h are randomly chosen generators of $\mathcal{G}$ and $\mathcal{H}$, respectively.
3. e is an efficiently computable bilinear map: $\mathcal{G} \times \mathcal{H} \to \mathcal{T}$, i.e., (1) for all $u, u' \in \mathcal{G}$ and $v, v' \in \mathcal{H}$, $e(uu', v) = e(u, v)e(u', v)$ and $e(u, vv') = e(u, v)e(u, v')$, and thus for all $u \in \mathcal{G}$, $v \in \mathcal{H}$ and $a, b \in Z$, $e(u^a, v^b) = e(u, v)^{ab}$, and (2) $e(g, h) \neq 1$.

We can set $\mathcal{G} = \mathcal{H}$, but we allow $\mathcal{G} \neq \mathcal{H}$ for the generality. Thus, our scheme can be implemented on both supersingular curves and ordinary curves. The bilinear map can be efficiently implemented with the Tate pairing (or the η_T pairing [3] on supersingular curves or the Ate pairing [19] on ordinary curves).

3.2 Assumptions

Our schemes are based on the q-SDH assumption [7, 8] on $\mathcal{G}$. Furthermore, our schemes adopt the tracing mechanism of [16], where, in addition to the bilinear groups, another group $\mathcal{F}$ with the same order p and the DDH assumption is required.

Definition 1 (q-SDH assumption). *For all PPT algorithm $\mathcal{A}$, the probability*

$$\Pr[\mathcal{A}(u, u^a, \ldots, u^{(a^q)}) = (b, u^{(1/a+b)}) \wedge b \in Z_p]$$

is negligible, where $u \in_R \mathcal{G}$ and $a \in_R Z_p$.

Definition 2 (DDH assumption). *For all PPT algorithm $\mathcal{A}$, the probability*

$$|\Pr[\mathcal{A}(u, u^a, u^b, u^{ab}) = 1] - \Pr[\mathcal{A}(u, u^a, u^b, u^c) = 1]|$$

is negligible, where $u \in_R \mathcal{F}$ and $a, b, c \in_R Z_p$.

Based on the q-SDH assumption, the DL (Discrete Logarithm) assumption also holds.

Definition 3 (DL assumption). *For all PPT algorithm $\mathcal{A}$, the probability*

$$\Pr[\mathcal{A}(u, u^a) = a]$$

is negligible, where $u \in_R \mathcal{G}$ and $a \in_R Z_p$.

3.3 BB Signatures

Our group signature schemes utilize Boneh-Boyen (BB) signature scheme [6]. As shown in [7], the knowledge of this signature (and the message) can be proved by a zero-knowledge proof for a representation, which is shown in Sec. 3.5.

BB-Setup: Select bilinear groups $\mathcal{G}, \mathcal{H}, \mathcal{T}$ with a prime order p and a bilinear map e. Select $g \in_R \mathcal{G}$ and $h \in_R \mathcal{H}$.
BB-KeyGen: Select $X \in_R Z_p$ and compute $Y = h^X$. The secret key is X and the public key is $(p, \mathcal{G}, \mathcal{H}, \mathcal{T}, e, g, h, Y)$.
BB-Sign: Given message $m \in Z_p$, compute $A = g^{1/(X+m)}$. The signature is A.
BB-Verify: Given message m and the signature A, check $e(A, Yh^m) = e(g, h)$.

BB signatures are existentially unforgeable against *weak* chosen message attack under the q-SDH assumption [6]. In this attack, the adversary must choose messages queried for the oracle, before the public key is given.

3.4 BBS+ Signatures

This signature scheme is an extension from BB signature scheme, and is informally introduced in [7], and the concrete construction is shown in [16, 1]. We call this signature BBS+ signature, as well as [1]. This scheme allows us to sign a set of messages. Also, the knowledge of the signature and messages can be proved [16, 1], as well as BB signatures.

BBS+-Setup: Select bilinear groups $\mathcal{G}, \mathcal{H}, \mathcal{T}$ with a prime order p and a bilinear map e. Select $g, g_1, \ldots, g_{L+1} \in_R \mathcal{G}$ and $h \in_R \mathcal{H}$.
BBS+-KeyGen: Select $X \in_R Z_p$ and compute $Y = h^X$. The secret key is X and the public key is $(p, \mathcal{G}, \mathcal{H}, \mathcal{T}, e, g, g_1, \ldots, g_{L+1}, h, Y)$.
BBS+-Sign: Given messages $m_1, \ldots, m_L \in Z_p$, select $y, z \in_R Z_p$ and compute

$$A = (g_1^{m_1} \cdots g_L^{m_L} g_{L+1}^y g)^{1/(X+z)}.$$

The signature is (A, y, z).
BBS+-Verify: Given messages $m_1, \ldots, m_L$ and the signature (A, y, z), check

$$e(A, Yh^z) = e(g_1^{m_1} \cdots g_L^{m_L} g_{L+1}^y g, h).$$

BBS+ signatures are existentially unforgeable against adaptively chosen message attack under the q-SDH assumption [2].

3.5 Proving Relations on Representations

As well as [7, 8, 16], 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. 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 $\mathcal{G}, \mathcal{T}, \mathcal{F}$ are utilized.

SPK **of representation:** An SPK proving the knowledge of a representation of $C \in \mathcal{G}$ to the bases $g_1, g_2, \ldots, g_t \in \mathcal{G}$ 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 groups $\mathcal{T}, \mathcal{F}$.

SPK **of representations with equal parts:** An SPK proving the knowledge of representations of $C, C' \in \mathcal{G}$ to the bases $g_1, \ldots, g_t \in \mathcal{G}$ 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 $\mathcal{G}, \mathcal{T}$ and $\mathcal{F}$ with the same order p, such as

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

where $C, g_1, \ldots, g_t \in \mathcal{G}$ and $C', h_1, \ldots, h_t \in \mathcal{T}$.

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 mechanism of conventional (non-revocable) group signature schemes is informally as follows. When a member joins, the member sends $f(x)$ to GM, where

f is a one-way function and x is a secret. GM returns a membership certificate $S = Sign(x)$ to the member, where $Sign$ is a signing function of GM. Then, the group signature consists of $E = Enc(f(x))$, where Enc is an encryption function using the manager's public key, and the following SPK on the signed message M.

$$SPK\{(x, S) : S = Sign(x) \wedge E = Enc(f(x))\}(M).$$

When opening the group signature, the manager decrypts E to check the sender of $f(x)$ in joining.

We borrow this mechanism from the Furukawa-Imai scheme [16], which is the one improved on the efficiency from [7] and currently is the most efficient pairing-based scheme. To this component, we add a novel revocation mechanism for realizing the constant computational complexity in signing/verifying.

The membership certificate in our scheme is modified to $S = Sign(x, UID)$, where UID is the ID of the member. On the other hand, a revocation list $\boldsymbol{RL}_t$ consists of

$$(Sign(t, RID_0, RID_1), \ldots, Sign(t, RID_r, RID_{r+1})).$$

RID_i $(1 \le i \le r)$ is the UID of a revoked member. Then, we can assume that RID_i is sorted such that $RID_i < RID_{i+1}$ for all $1 \le i \le r$. In addition, RID_0 and RID_{r+1} are special IDs, where $RID_0 < UID$ and $RID_{r+1} > UID$ for any UID.

The group signature for $\boldsymbol{RL}_t$ is the following SPK.

$$\begin{aligned}&SPK\{(x, UID, S, RID_i, RID_{i+1}, \tilde{S}) : \\ &\quad S = Sign(x, UID) \wedge E = Enc(f(x)) \\ &\quad \wedge \tilde{S} = Sign(t, RID_i, RID_{i+1}) \\ &\quad \wedge RID_i < UID \wedge UID < RID_{i+1}\}(M).\end{aligned}$$

Clearly any non-revoked member can prove this SPK. On the other hand, if the member with UID is revoked, $UID = RID_{\tilde{\imath}}$ holds for some $\tilde{\imath}$. Then, $RID_i < UID$ holds for all $i < \tilde{\imath}$, and $UID < RID_i$ for all $i > \tilde{\imath}$. Thus, the revoked member cannot find i such that $RID_i < UID < RID_{i+1}$, which means that the member cannot prove this SPK, since the correctness of UID, RID_i and RID_{i+1} are also ensured by the certificates S and $\tilde{S}$ at the current time t.

The costs of the SPKs for inequations have $O(1)$ complexity, as the construction idea of the SPK is shown later. Thus, the computational costs for signing/verifying are $O(1)$ w.r.t. R and N. The size of a signature is also $O(1)$. The size of $\boldsymbol{RL}_t$ is $O(R)$. The overhead is the revocation complexity of GM, that is, each revocation requires $O(R)$ computation, and the long public key with $O(N)$ size due to the following SPK for inequations.

Remark 1. Our novel idea is to use the above integer inequations on UID and RID's. Instead, we easily get a simple solution of proving that his UID is not equal to **all** RID's, but it requires $O(R)$ complexity. Camenisch and Lysyanskaya proposed an elegant idea of the dynamic accumulator [14], but it needs the incremental update of signer's secret key. On the other hand, in our solution, the signer has only to prove the inequations for **some** i, and thus has $O(1)$ complexity. And, note that it does not need any update of the secret key. As far as we know, such a solution is unknown, and thus our construction idea has a sufficient novelty together with the practicality on efficiency.

Remark 2. Note that this group signature does not reveal any information on i, RID_i and RID_{i+1} to the verifier. This is because the SPK proves only the fact that there are secrets i, RID_i and RID_{i+1} satisfying the inequations $RID_i < UID < RID_{i+1}$, without revealing i, RID_i and RID_{i+1}.

Proving Integer Inequations. In the above construction idea, we need an efficient SPK proving an integer inequation on secrets, as $RID_i < UID$. In the strong RSA setting, Boudot's SPK [9] can be used. However, this methodology cannot be easily adopted to the pairing-based setting. On the other hand, Teranishi and Sako utilize an SPK [25] proving that a secret is in an integer interval, and the SPK is effective even in the pairing-based setting. We adopt this SPK and extend it to the SPK proving the integer inequation.

At first, we consider the SPK proving that a secret w is in the interval $[1, N]$. This SPK needs a special setup, where the trusted party (GM in our setting) issues certificates. The certificate is a BB signature on every element from the interval $[1, N]$, as $Sign(1), \ldots, Sign(N)$. These certificates are given to each prover, as a part of the public key. Then, the SPK proving $w \in [1, N]$ is computed as

$$SPK\{(w, S') : S' = Sign(w)\}(M).$$

Since issued certificates are for only $1, \ldots, N$, it ensures $w \in [1, N]$.

Next, using this SPK, we consider the SPK proving $y > x$, in the situation that $x, y \in [1, N]$ is ensured (Note that it is ensured in the above group signature, due to $S = Sign(x, UID) \wedge \tilde{S} = Sign(t, RID_i, RID_{i+1})$). The SPK is obtained as

$$SPK\{(x, y, S') : S' = Sign(y - x \bmod p)\}(M).$$

Then, this ensures $z \in [1, N]$, where $z = y - x \pmod p$. On the other hand, from $x, y \in [1, N]$, we obtain $z \in [1, N]$ when $y > x$, and $z \in [p - N, p - 1]$ or $z = 0$ when $y \leq x$. Since we can assume $N < p/2$, $z \in [1, N]$ means $y > x$.

The computational cost of this SPK is constant w.r.t. N, although the distributions of N certificates is an overhead.

Remark 3. As another approach of the SPK proving the interval relation, a bit-by-bit approach is also known, which is described and used in [1]. However, in this approach, the size of the proof is linear in the size of the secret to be proved. In case of adopting it in our group signatures, the size is $\lceil \log_2 N \rceil$, and the proof size is about $4 \lceil \log_2 N \rceil$ multiplied by the size of the group element. If we use the same parameters as Sect. 7 and $N = 1,000,000$, the SPK amounts to more than 2,000 Bytes. Since we use interval proofs two times in our group signatures, the group signature is about more than 4,000 Bytes. Even in case of $N = 1,000$, it is more than 2,000 Bytes. On the other hand, our group signature proposed in this section is about 650 Bytes, and our extended group signature proposed in Sect. 6 is about 1,200 Bytes.

In the bit-by-bit approach, the public key size is constant and better. However, signatures frequently occur in authentications or signings, and thus, in lots of applications, the signature size influences the efficiency of communication and storage more than the size of the public key that is not changed. This is why we adopt Teranishi-Sako's SPK proving interval relations in this paper.

4.2 Proposed Algorithms

Assume that the total number of group members, N, is fixed in advance, and we can assume that $N < p/2$.

Setup: The input of this algorithm is security parameter 1^ℓ, and the output is $param$.

1. Select bilinear groups $\mathcal{G}, \mathcal{H}, \mathcal{T}$ with the same prime order p of length ℓ, and the bilinear map e. In addition, select a group $\mathcal{F}$ with the DDH assumption and the same prime order p. Select hash function $H : \{0,1\}^* \to Z_p$.
2. Select $g, g_1, g_2, g_3, \tilde{g}, \tilde{g}_1, \hat{g}, \hat{g}_1, \hat{g}_2, \hat{g}_3, \hat{g}_4 \in_R \mathcal{G}$, $h, \tilde{h}, \hat{h} \in_R \mathcal{H}$, and $f \in_R \mathcal{F}$.
3. Output $param = (p, \mathcal{G}, \mathcal{H}, \mathcal{T}, \mathcal{F}, e, H, g, g_1, g_2, g_3, \tilde{g}, \tilde{g}_1, \hat{g}, \hat{g}_1, \hat{g}_2, \hat{g}_3, \hat{g}_4, h, \tilde{h}, \hat{h}, f)$.

KeyGen: The inputs of this algorithm are N and $param = (p, \mathcal{G}, \mathcal{H}, \mathcal{T}, \mathcal{F}, e, H, g, g_1, g_2, g_3, \tilde{g}, \tilde{g}_1, \hat{g}, \hat{g}_1, \hat{g}_2, \hat{g}_3, \hat{g}_4, h, \tilde{h}, \hat{h}, f)$, and the output consists of gpk and msk.

1. Select $X, \tilde{X}, \hat{X} \in_R Z_p$ and compute $Y = h^X$, $\tilde{Y} = \tilde{h}^{\tilde{X}}$, and $\hat{Y} = \hat{h}^{\hat{X}}$.
2. Select $X_1, X_2 \in_R Z_p$ and compute $Y_1 = f^{X_1}$ and $Y_2 = f^{X_2}$.
3. For all $j \in 1, \ldots, N$, generate BB signatures for j, namely compute $F_j = \tilde{g}^{1/(\tilde{X}+j)}$.
4. Output $gpk = (p, \mathcal{G}, \mathcal{H}, \mathcal{T}, \mathcal{F}, e, H, g, g_1, g_2, g_3, \tilde{g}, \tilde{g}_1, \hat{g}, \hat{g}_1, \hat{g}_2, \hat{g}_3, \hat{g}_4, h, \tilde{h}, \hat{h}, f, Y, \tilde{Y}, \hat{Y}, Y_1, Y_2, F_1, \ldots F_N)$ and $msk = (X, \tilde{X}, \hat{X}, X_1, X_2)$.

Join: This is an interactive protocol between **Join-U** (the i-th joining user) and **Join-GM** (GM). The common input is gpk, and the input of **Join-GM** is msk. The output of **Join-U** is $\boldsymbol{usk}[i]$. The output is **Join-GM** is $\boldsymbol{reg}[i]$. Assume that $i = 1$ and $i = N$ are assigned to fictitious users out of the group (Assume that the users of $i = 1, N$ are always revoked).

1. [**Join-U**] Select $x_i, y'_i \in Z_p$, compute $A'_i = g_1^{x_i} g_3^{y'_i}$ and $D_i = f^{x_i}$, and send A'_i, D_i to **Join-GM**. In addition, prove the validity of A'_i and D_i using an SPK for representations.

2. [**Join-GM**] Select $y_i'', z_i \in_R Z_p$, compute
$$A_i = (A_i' g_2^i g_3^{y_i''} g)^{1/(X+z_i)},$$
and return (i, A_i, y_i'', z_i) to **Join-U**. Output $\boldsymbol{reg}[i] = D_i$.
3. [**Join-U**] Compute $y_i = y_i' + y_i'' \bmod p$, verify $e(A_i, Yh^{z_i}) = e(g_1^{x_i} g_2^i g_3^{y_i} g, h)$, and output $\boldsymbol{usk}[i] = (A_i, x_i, y_i, z_i)$ s.t. $A_i^{X+z_i} = g_1^{x_i} g_2^i g_3^{y_i} g$. The BBS+ signatures (A_i, y_i, z_i) on secret x_i and UID i is correspondent to the membership certificate.

Revoke: The input of this algorithm consists of gpk, t and RU. The output is $\boldsymbol{RL}_t$.

1. Sort elements of RU, according to ascending order. Let $\hat{\imath}_1, \ldots, \hat{\imath}_r$ be the sorted ones, where $r = |RU|$. In addition, set $\hat{\imath}_0 = 1$ and $\hat{\imath}_{r+1} = N$.
2. For every $\hat{\imath}_j$ $(0 \le j \le r)$, generate a BBS+ signature on $t, \hat{\imath}_j, \hat{\imath}_{j+1}$, namely select $\hat{y}_j, \hat{z}_j \in_R Z_p$, and compute $B_{\hat{\imath}_j} = (\hat{g}_1^t \hat{g}_2^{\hat{\imath}_j} \hat{g}_3^{\hat{\imath}_{j+1}} \hat{g}_4^{\hat{y}_j} \hat{g})^{1/(\hat{X}+\hat{z}_j)}$.
3. Output
$$\boldsymbol{RL}_t = ((\hat{\imath}_0, \hat{\imath}_1, B_{\hat{\imath}_0}, \hat{y}_0, \hat{z}_0), \ldots, (\hat{\imath}_r, \hat{\imath}_{r+1}, B_{\hat{\imath}_r}, \hat{y}_r, \hat{z}_r)).$$

Sign: The input of this algorithm consists of gpk, $\boldsymbol{usk}[i] = (A_i, x_i, y_i, z_i)$, t, $\boldsymbol{RL}_t = ((\hat{\imath}_0, \hat{\imath}_1, B_{\hat{\imath}_0}, \hat{y}_0, \hat{z}_0), \ldots, (\hat{\imath}_r, \hat{\imath}_{r+1}, B_{\hat{\imath}_r}, \hat{y}_r, \hat{z}_r))$ and $M \in \{0,1\}^*$. The output is σ.

1. Select a random $\alpha \in_R Z_p$, and compute a commitment $C = A_i g_3^{\alpha}$. Set $\zeta = y_i + \alpha z_i$.
2. Find $\hat{\imath}_j$ s.t. $\hat{\imath}_j < i < \hat{\imath}_{j+1}$. Select a random $\hat{\alpha} \in_R Z_p$, and compute a commitment $\hat{C} = B_{\hat{\imath}_j} \hat{g}_4^{\hat{\alpha}}$. Set $\hat{\zeta} = \hat{y}_j + \hat{\alpha} \hat{z}_j$.
3. Set $\delta_1 = i - \hat{\imath}_j$ and $\delta_2 = \hat{\imath}_{j+1} - i$. Find F_{δ_1} and F_{δ_2}, select $\beta_1, \beta_2 \in_R Z_p$, and compute commitments $C_{F_{\delta_1}} = F_{\delta_1} \tilde{g}_1^{\beta_1}$ and $C_{F_{\delta_2}} = F_{\delta_2} \tilde{g}_1^{\beta_2}$. Set $\theta_1 = \beta_1 \delta_1$, and $\theta_2 = \beta_2 \delta_2$.
4. Select a random $\gamma \in_R Z_p$, and compute ciphertext $T_1 = f^{x_i+\gamma}$, $T_2 = Y_1^{\gamma}$, and $T_3 = Y_2^{\gamma}$.
5. Compute an SPK V on message M proving knowledge of $x_i, i, \zeta, \alpha, z_i, \hat{\imath}_j, \hat{\imath}_{j+1}, \hat{\zeta}, \hat{\alpha}, \hat{z}_j, \theta_1, \beta_1, \theta_2, \beta_2, \gamma$ s.t.
$$\begin{aligned}
&e(C, Y)e(g, h)^{-1} = e(g_1, h)^{x_i} e(g_2, h)^i e(g_3, h)^{\zeta} e(g_3, Y)^{\alpha} e(C, h)^{-z_i},\\
&e(\hat{C}, \hat{Y})e(\hat{g}, \hat{h})^{-1} e(\hat{g}_1, \hat{h})^{-t}\\
&\qquad = e(\hat{g}_2, \hat{h})^{\hat{\imath}_j} e(\hat{g}_3, \hat{h})^{\hat{\imath}_{j+1}} e(\hat{g}_4, \hat{h})^{\hat{\zeta}} e(\hat{g}_4, \hat{Y})^{\hat{\alpha}} e(\hat{C}, \hat{h})^{-\hat{z}_j},\\
&e(C_{F_{\delta_1}}, \tilde{Y})e(\tilde{g}, \tilde{h})^{-1} = e(\tilde{g}_1, \tilde{h})^{\theta_1} e(\tilde{g}_1, \tilde{Y})^{\beta_1} e(C_{F_{\delta_1}}, \tilde{h})^{-(i-\hat{\imath}_j)},\\
&e(C_{F_{\delta_2}}, \tilde{Y})e(\tilde{g}, \tilde{h})^{-1} = e(\tilde{g}_1, \tilde{h})^{\theta_2} e(\tilde{g}_1, \tilde{Y})^{\beta_2} e(C_{F_{\delta_2}}, \tilde{h})^{-(\hat{\imath}_{j+1}-i)},\\
&T_1 = f^{x_i+\gamma}, \quad T_2 = Y_1^{\gamma}, \quad T_3 = Y_2^{\gamma}.
\end{aligned}$$
As indicated in Lemma 1, the above relations in V ensure the validity of signatures (A_i, y_i, z_i), $(B_{\hat{\imath}_j}, \hat{y}_j, \hat{z}_j)$, F_{δ_1}, F_{δ_2} and the validity of T_1, T_2 and T_3, respectively.
6. Output $\sigma = (C, \hat{C}, C_{F_{\delta_1}}, C_{F_{\delta_2}}, T_1, T_2, T_3, V)$.

Verify: The inputs are gpk, t, a target signature $\sigma = (C, \hat{C}, C_{F_{\delta_1}}, C_{F_{\delta_2}}, T_1, T_2, T_3, V)$, and the message M. Check the SPK V. Output 'valid' (resp., 'invalid') if it is correct (resp., incorrect).

Open: The inputs are gpk, the secret key $msk = (X, \tilde{X}, \hat{X}, X_1, X_2)$, $\boldsymbol{reg}$ with $\boldsymbol{reg}[i] = D_i$, t, a target signature $\sigma = (C, \hat{C}, C_{F_{\delta_1}}, C_{F_{\delta_2}}, T_1, T_2, T_3, V)$ and the message M.

1. Verify σ. If it is invalid, abort.
2. Using X_1, compute $T_1/T_2^{1/X_1}$ to obtain f^{x_i}. Search $\boldsymbol{reg}$ for i with $D_i = f^{x_i}$.
3. Output i.

5 Security

Here, we show a lemma and theorems on the security of our scheme.

Lemma 1. *The SPK V proves the knowledge of $x_i, i, z_i, y_i, \hat{\imath}_j, \hat{\imath}_{j+1}, \hat{z}_j, \hat{y}_j, \eta_1, \eta_2, \gamma, A_i, B_{\hat{\imath}_j}, F'_{\delta_1}, F'_{\delta_2}$ s.t.*

$$A_i = (g_1^{x_i} g_2^{i} g_3^{y_i} g)^{1/(X+z_i)}, \qquad B_{\hat{\imath}_j} = (\hat{g}_1^{t} \hat{g}_2^{\hat{\imath}_j} \hat{g}_3^{\hat{\imath}_{j+1}} \hat{g}_4^{\hat{y}_j} \hat{g})^{1/(\hat{X}+\hat{z}_j)},$$
$$F'_{\delta_1} = (\tilde{g}_1^{\eta_1} \tilde{g})^{1/(\tilde{X}+(i-\hat{\imath}_j))}, \qquad F'_{\delta_2} = (\tilde{g}_1^{\eta_2} \tilde{g})^{1/(\tilde{X}+(\hat{\imath}_{j+1}-i))},$$
$$T_1 = f^{x_i+\gamma}, \qquad T_2 = Y_1^{\gamma}, \qquad T_3 = Y_2^{\gamma}.$$

This proof will be in the full paper. Note that F'_{δ_1} and F'_{δ_2} are variants of BB signatures, and are not the same as F_{δ_1} and F_{δ_2}, due to the parts $\tilde{g}_1^{\eta_1}, \tilde{g}_1^{\eta_2}$. However, in the traceability game, the difference can be treated well. Note that, as well as [21], the adopted SPKs of F'_{δ_1} and F'_{δ_2} are more efficient than those of F_{δ_1} and F_{δ_2} described in [7].

Theorem 1. *The proposed scheme satisfies the traceability under the q-SDH assumption, in the random oracle model.*

This proof will be in the full paper. In the proof, if an adversary wins the traceability game for our scheme, using this adversary, we can forge BBS+ signatures (A_i, y_i, z_i) or $(B_{\hat{\imath}_j}, \hat{y}_j, \hat{z}_j)$, or a BB signature F_k. Thus, we can construct adversaries for the BBS+ signatures or BB+ signatures, which are secure under the q-SDH assumption.

Theorem 2. *The proposed scheme satisfies the anonymity under the DDH assumption, in the random oracle model.*

This proof is similar to [17], which is the full-paper version of [16]. As well as [16], our group signature consists of a double encryption of an ID f^{x_i}, (T_1, T_2, T_3), and the non-interactive zero-knowledge proof including statistically hiding commitments, which mean an IND-CCA2 secure encryption. Thus, easily we can reduce the anonymity game of our scheme to the IND-CCA2 game for the IND-CCA2 secure encryption under the DDH assumption.

Theorem 3. *The proposed scheme satisfies the non-frameability under the DL assumption, in the random oracle model.*

This proof is also similar to [17]. In this proof, the DL adversary is constructed using the adversary in the non-frameability game. In the game, instead of computing (f, f^{x_i}), the input of the DL adversary is used as (f, f^{x_i}). In an honest user's joining, f^{x_i} in the input (x_i is unknown) is used. The SPKs for x_i in joining and signing can be simulated by the zero-knowledge simulator. From the output of the adversary in the non-frameability game (which outputs the signature for the honest user with x_i with a non-negligible probability), we can extract x_i, which means breaking the DL assumption.

6 Extension

The weak point of the proposed scheme is $O(N)$ size of gpk. This section shows an extended scheme with $O(\sqrt{N})$-size public key.

6.1 Idea

The extension is obtained by improving the SPK proving integer inequation. A positive integer $w \in [1, N]$ can be expressed as $w_1^2 + w_2$, where w_1 is the greatest square less than w, and w_2 is a non-negative integer less than $2\sqrt{N}$ [9]. Then, note that $1 \le w_1 < \sqrt{N}$ and $0 \le w_2 < 2\sqrt{N}$. The extended integer inequation proof is as follows. Define $N_1 = \lfloor \sqrt{N} \rfloor$ and $N_2 = \lfloor 2\sqrt{N} \rfloor$.

In the setup, the trusted party issues two types of certificates based on BB signatures. In one type, $Sign(1), \ldots, Sign(N_1)$ are issued. In the other, $Sign'(0), \ldots, Sign'(N_2)$ are issued.

Then, the SPK proving $y > x$ is computed as

$$SPK\{(x, y, w_1, w_2, S_1, S_2) : y - x = w_1^2 + w_2 \pmod p \\ \wedge S_1 = Sign(w_1) \wedge S_2 = Sign'(w_2)\}(M).$$

The relation $y - x = w_1^2 + w_2 \pmod p$ can be efficiently proved by an SPK, as [9]. Due to the BB signatures, $w_1 \in [1, N_1]$ and $w_2 \in [0, N_2]$ are ensured. Then, $w_1^2 + w_2 \in [1, N_1^2 + N_2]$. By assuming $N_1^2 + N_2 < p/2$, we obtain $w_1^2 + w \in [1, p/2 - 1]$. Namely, for $z = y - x \pmod p$, we have $z \in [1, p/2 - 1]$. Since $x, y \in [1, N]$ is ensured, as well as the basic scheme, this means $y - x > 0$ and thus $y > x$ in Z.

On the other hand, the number of the issued certificates is $O(\sqrt{N})$, which means $O(\sqrt{N})$-size public key.

6.2 Extended Algorithms

Define $N_1 = \lfloor \sqrt{N} \rfloor$ and $N_2 = \lfloor 2\sqrt{N} \rfloor$. We assume that $N_1^2 + N_2 < p/2$. **Revoke**, **Verify**, and **Open** are similar to the basic scheme in Sec. 4. The others are modified as follows.

Setup: In addition to **Setup** of the basic scheme, select $\dot{g}, \dot{g}_1 \in \mathcal{G}, \dot{h} \in \mathcal{H}$, which are added to *param*.

KeyGen: In addition to Step 1 and 2 of **KeyGen** of the basic scheme, select $\dot{X} \in_R Z_p$, and compute $\dot{Y} = \dot{h}^{\dot{X}}$. Step 3 is modified as follows.

3. For all $j \in 1, \ldots, N_1$, generate BB signatures for j, namely compute $F_j = \tilde{g}^{1/(\tilde{X}+j)}$. Additionally, for all $j \in 0, \ldots, N_2$, generate BB signatures for j, namely compute $\dot{F}_j = \dot{g}^{1/(\dot{X}+j)}$.

Add $\dot{X}$ to *msk*, and add $\dot{Y}$, $F_1, \ldots, F_{N_1}$, $\dot{F}_0, \ldots, \dot{F}_{N_2}$ to *gpk*.

Sign: Step 1, 2, 4 are the same as the basic scheme. Step 3, 5 are modified as follows.

3. Set $\delta_1 = i - \hat{\imath}_j$ and $\delta_2 = \hat{\imath}_{j+1} - i$. Find $\delta_{1,1}$ and $\delta_{1,2}$ s.t. $\delta_1 = {\delta_{1,1}}^2 + \delta_{1,2}$, $1 \le \delta_{1,1} \le N_1$ and $0 \le \delta_{1,2} \le N_2$. Find $\delta_{2,1}$ and $\delta_{2,2}$ s.t. $\delta_2 = {\delta_{2,1}}^2 + \delta_{2,2}$, $1 \le \delta_{2,1} \le N_1$ and $0 \le \delta_{2,2} \le N_2$. Find $F_{\delta_{1,1}}$, $\dot{F}_{\delta_{1,2}}$, $F_{\delta_{2,1}}$, and $\dot{F}_{\delta_{2,2}}$, select $\beta_{1,1}, \beta_{1,2}, \beta_{2,1}, \beta_{2,2} \in_R Z_p$, and compute commitments $C_{F_{\delta_{1,1}}} = F_{\delta_{1,1}} \tilde{g}_1^{\beta_{1,1}}$, $C_{\dot{F}_{\delta_{1,2}}} = \dot{F}_{\delta_{1,2}} \dot{g}_1^{\beta_{1,2}}$, $C_{F_{\delta_{2,1}}} = F_{\delta_{2,1}} \tilde{g}_1^{\beta_{2,1}}$, $C_{\dot{F}_{\delta_{2,2}}} = \dot{F}_{\delta_{2,2}} \dot{g}_1^{\beta_{2,2}}$. Set $\theta_{1,1} = \beta_{1,1}\delta_{1,1}$, $\theta_{1,2} = \beta_{1,2}\delta_{1,2}$, $\theta_{2,1} = \beta_{2,1}\delta_{2,1}$, and $\theta_{2,2} = \beta_{2,2}\delta_{2,2}$.
   Furthermore, select $\xi_1, \xi_1', \xi_2, \xi_2' \in_R Z_p$, and compute commitments $C_{\delta_{1,1}} = \tilde{g}^{\delta_{1,1}} {\tilde{g}_1}^{\xi_1}$, $C_{{\delta_{1,1}}^2} = \tilde{g}^{{\delta_{1,1}}^2} {\tilde{g}_1}^{\xi_1'}$, $C_{\delta_{2,1}} = \tilde{g}^{\delta_{2,1}} {\tilde{g}_1}^{\xi_2}$, $C_{{\delta_{2,1}}^2} = \tilde{g}^{{\delta_{2,1}}^2} {\tilde{g}_1}^{\xi_2'}$. Set $\xi_1'' = \xi_1' - \xi_1\delta_{1,1}$ and $\xi_2'' = \xi_2' - \xi_2\delta_{2,1}$.
5. Compute an $SPK\ V$ on message M proving knowledge of $x_i, i, \zeta, \alpha, z_i, \hat{\imath}_j, \hat{\imath}_{j+1}, \hat{\zeta}, \hat{\alpha}, \hat{z}_j, \delta_{1,1}, \delta_{1,2}, \delta_{2,1}, \delta_{2,2}, \theta_{1,1}, \theta_{1,2}, \theta_{2,1}, \theta_{2,2}, \beta_{1,1}, \beta_{1,2}, \beta_{2,1}, \beta_{2,2}, \xi_1, \xi_1', \xi_1'', \xi_2, \xi_2', \xi_2'', \gamma$ s.t.

$$\begin{aligned}
&e(C,Y)e(g,h)^{-1} = e(g_1,h)^{x_i} e(g_2,h)^{i} e(g_3,h)^{\zeta} e(g_3,Y)^{\alpha} e(C,h)^{-z_i},\\
&e(\hat{C},\hat{Y})e(\hat{g},\hat{h})^{-1}e(\hat{g}_1,\hat{h})^{-t}\\
&\qquad\qquad = e(\hat{g}_2,\hat{h})^{\hat{\imath}_j} e(\hat{g}_3,\hat{h})^{\hat{\imath}_{j+1}} e(\hat{g}_4,\hat{h})^{\hat{\zeta}} e(\hat{g}_4,\hat{Y})^{\hat{\alpha}} e(\hat{C},\hat{h})^{-\hat{z}_j},\\
&e(C_{F_{\delta_{1,1}}},\tilde{Y})e(\tilde{g},\tilde{h})^{-1} = e(\tilde{g}_1,\tilde{h})^{\theta_{1,1}} e(\tilde{g}_1,\tilde{Y})^{\beta_{1,1}} e(C_{F_{\delta_{1,1}}},\tilde{h})^{-\delta_{1,1}},\\
&e(C_{\dot{F}_{\delta_{1,2}}},\dot{Y})e(\dot{g},\dot{h})^{-1} = e(\dot{g}_1,\dot{h})^{\theta_{1,2}} e(\dot{g}_1,\dot{Y})^{\beta_{1,2}} e(C_{\dot{F}_{\delta_{1,2}}},\dot{h})^{-\delta_{1,2}},\\
&e(C_{F_{\delta_{2,1}}},\tilde{Y})e(\tilde{g},\tilde{h})^{-1} = e(\tilde{g}_1,\tilde{h})^{\theta_{2,1}} e(\tilde{g}_1,\tilde{Y})^{\beta_{2,1}} e(C_{F_{\delta_{2,1}}},\tilde{h})^{-\delta_{2,1}},\\
&e(C_{\dot{F}_{\delta_{2,2}}},\dot{Y})e(\dot{g},\dot{h})^{-1} = e(\dot{g}_1,\dot{h})^{\theta_{2,2}} e(\dot{g}_1,\dot{Y})^{\beta_{2,2}} e(C_{\dot{F}_{\delta_{2,2}}},\dot{h})^{-\delta_{2,2}},\\
&C_{\delta_{1,1}} = \tilde{g}^{\delta_{1,1}} {\tilde{g}_1}^{\xi_1},\quad C_{{\delta_{1,1}}^2} = C_{\delta_{1,1}}^{\delta_{1,1}} {\tilde{g}_1}^{\xi_1''},\quad C_{{\delta_{1,1}}^2} = \tilde{g}^{-\delta_{1,2}+(i-\hat{\imath}_j)} {\tilde{g}_1}^{\xi_1'},\\
&C_{\delta_{2,1}} = \tilde{g}^{\delta_{2,1}} {\tilde{g}_1}^{\xi_2},\quad C_{{\delta_{2,1}}^2} = C_{\delta_{2,1}}^{\delta_{2,1}} {\tilde{g}_1}^{\xi_2''},\quad C_{{\delta_{2,1}}^2} = \tilde{g}^{-\delta_{2,2}+(\hat{\imath}_{j+1}-i)} {\tilde{g}_1}^{\xi_2'},\\
&T_1 = f^{x_i+\gamma},\quad T_2 = Y_1^{\gamma},\quad T_3 = Y_2^{\gamma}.
\end{aligned}$$

Security. The proof sketch of the traceability will also be in the full paper, which is similar to the proof of the basic scheme. The anonymity and non-frameability can be proved as well as the basic scheme.

7 Efficiency

Computational efficiency. At first, we discuss the efficiency of the basic scheme in Sec. 4. As well as [22], all pairings in the **Sign** algorithm can be pre-computed, and pairings except 4 pairing in the **Verify** algorithm can be also pre-computed. Then, the **Sign** algorithm requires 4 exponentiations on $\mathcal{G}$, 6 exponentiations on $\mathcal{F}$, and 4 multi-exponentiations on $\mathcal{T}$. The **Verify** algorithm requires 3 exponentiations on $\mathcal{F}$, 4 multi-exponentiations on $\mathcal{H}$, 4 multi-exponentiations on $\mathcal{T}$, and 4 pairings. The computational time of each exponentiation or each pairing does not depend on N and R, and thus constant computational costs for both **Sign** and **Verify** are achieved.

The next is the efficiency of the extended scheme in Sec. 6. By adopting the same pre-computations, the **Sign** algorithm requires 16 multi-exponentiations on $\mathcal{G}$, 6 exponentiations on $\mathcal{F}$, and 6 multi-exponentiations on $\mathcal{T}$. The **Verify** algorithm requires 6 multi-exponentiations on $\mathcal{G}$, 3 exponentiations on $\mathcal{F}$, 6 multi-exponentiations on $\mathcal{H}$, 6 multi-exponentiations on $\mathcal{T}$, and 6 pairings. Thus, we achieve $O(1)$ computational costs in signing/verifying in both schemes, although the extended one has some overheads for obtaining $O(\sqrt{N})$-size public key.

Data size. Here, we discuss the data size. To confirm the practicality, we use the following concrete parameters. To obtain the 112-bit security level, we can represent $\mathcal{G}$- and $\mathcal{F}$-elements with 224 bits for the ECC DL security. We assume the BN curves [4] with efficient pairing computations and the embedding degree 12. Then, we can represent $\mathcal{T}$-elements with 2688 bits, which satisfies the DL security corresponding the 112-bit security level.

Note that both schemes have signatures with constant sizes. In the above concrete setting, the signature of the basic scheme is about 650 Bytes. This is because the signature σ has 4 $\mathcal{G}$-elements, 3 $\mathcal{F}$-elements, and 16 Z_p-elements. On the other hand, the signature of the extended scheme is about 1,200 Bytes, since the signature has 10 $\mathcal{G}$-elements, 3 $\mathcal{F}$-elements, and 30 Z_p-elements.

Next, we discuss the public key size. In the basic scheme, the length of *gpk* is $O(N)$, due to the dominant $F_1, \ldots, F_N$. On the other hand, in the extended one, it is reduced to $O(\sqrt{N})$, due to the dominant $F_1, \ldots, F_{N_1}$, $\dot{F}_0, \ldots, \dot{F}_{N_2}$. To confirm the practicality of the extended one, we also use the above concrete parameters. Then, in case of $N = 1,000,000$, the public parameters $F_1, \ldots, F_{N_1}, \dot{F}_0, \ldots$, $\dot{F}_{N_2}$ only need about 84 KBytes in total. This concrete size shows the sufficient practicality of the storage, not only in usual PCs but also in smart phones. Furthermore, since clients have only to download the public key once, the communication cost does not matter.

As the final remark, in both schemes, the length of $\boldsymbol{RL}_t$ is $O(R)$.

8 Conclusion

In this paper, we have proposed revocable group signature schemes, where both signing and verifying require only constant computational costs w.r.t. the group size and the number of revoked members. In the schemes, any secret key update is not required, and the data related to revocation has $O(R)$ size.

One of our future works is to integrate this scheme into anonymous client authentications in WEB services, and to evaluate it in practical environments. Other future works are to decrease the size of the revocation list and to exclude the random oracle.

Acknowledgments

This work was supported by the Strategic Information and Communications R&D Promotion Programme (SCOPE) from the Ministry of Internal Affairs and Communications of Japan. We would like to thank the anonymous reviewers and Yasuyuki Nogami for helpful comments.

References

1. M.H. Au, W. Susilo, and Y. Mu, "Constant-size dynamic k-TAA," Security in Communication Networks: 5th International Conference, SCN 2006, LNCS 4116, pp.111–125, Springer–Verlag, 2006.
2. M.H. Au, W. Susilo, and Y. Mu, "Constant-size dynamic k-TAA." Cryptology ePrint Archive: Report 2008/136, 2008. This is the extended version of [1].
3. P.S.L.M. Barreto, S.D. Galbraith, C. O'hEigeartaigh, and M. Scott, "Efficient pairing computation on supersingular abelian varieties," Designs, Codes and Cryptography, vol.42, no.3, pp.239–271, 2007.
4. P.S.L.M. Barreto and M. Naehrig, "Pairing-friendly elliptic curves of prime order," Proc. 12th International Workshop on Selected Areas in Cryptography (SAC 2005), LNCS 3897, pp.319–331, Springer–Verlag, 2005.
5. M. Bellare, H. Shi, and C. Zhang, "Foundations of group signatures: The case of dynamic groups," Topics in Cryptology - CT-RSA 2005, LNCS 3376, pp.136–153, Springer–Verlag, 2005.
6. D. Boneh and X. Boyen, "Short signatures without random oracles," Advances in Cryptology — EUROCRYPT 2004, LNCS 3027, pp.56–73, Springer–Verlag, 2004.
7. D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," Advances in Cryptology — CRYPTO 2004, LNCS 3152, pp.41–55, Springer–Verlag, 2004.
8. 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.
9. F. Boudot, "Efficient proofs that a committed number lies in an interval," Advances in Cryptology — EUROCRYPT 2000, LNCS 1807, pp.431–444, Springer–Verlag, 2000.
10. E. Bresson and J. Stern, "Group signature scheme with efficient revocation," Proc. 4th International Workshop on Practice and Theory in Public Key Cryptography (PKC 2001), LNCS 1992, pp.190–206, Springer–Verlag, 2001.
11. E.F. Brickell, J. Camenisch, and L. Chen, "Direct anonymous attestation," Proc. 11th ACM Conference on Computer and Communications Security (ACM-CCS '04), pp.132–145, 2004.

12. J. Camenisch and J. Groth, "Group signatures: Better efficiency and new theoretical aspects," Security in Communication Networks: 4th International Conference, SCN 2004, LNCS 3352, pp.120–133, Springer–Verlag, 2005.
13. J. Camenisch and E.V. Herreweghen, "Design and implementation of the idemix anonymous credential system," Proc. 9th ACM Conference on Computer and Communications Security (ACM-CCS '02), pp.21–30, 2002.
14. 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.
15. D. Chaum and E. van Heijst, "Group signatures," Advances in Cryptology — EUROCRYPT '91, LNCS 547, pp.241–246, Springer–Verlag, 1991.
16. J. Furukawa and H. Imai, "An efficient group signature scheme from bilinear maps," Proc. 10th Australasian Conference on Information Security and Privacy (ACISP 2005), LNCS 3574, pp.455–467, Springer–Verlag, 2005.
17. J. Furukawa and H. Imai, "An efficient group signature scheme from bilinear maps," IEICE Trans. Fundamentals, vol.E89-A, no.5, pp.1328–1338, 2006.
18. J. Groth, "Fully anonymous group signatures without random oracles," Advances in Cryptology - ASIACRYPT 2007, LNCS 4833, pp.164–180, Springer–Verlag, 2007.
19. F. Hess, N. Smart, and F. Vercauteren, "The eta pairing revisited," IEEE Trans. Information Theory, vol.52, no.10, pp.4595–4602, 2006.
20. T. Isshiki, K. Mori, K. Sako, I. Teranishi, and S. Yonezawa, "Using group signatures for identity management and its implementation," Proc. 2nd ACM Workshop on Digital Identity Management, pp.73–78, 2006.
21. T. Nakanishi and N. Funabiki, "A short verifier-local revocation group signature scheme with backward unlinkability," Proc. First International Workshop on Security (IWSEC 2006), LNCS 4266, pp.17–32, Springer–Verlag, 2006.
22. T. Nakanishi and N. Funabiki, "Short verifier-local revocation group signature scheme with backward unlinkability," IEICE Trans. Fundamentals, vol.E90-A, no.9, pp.1793–1802, 2007.
23. T. Nakanishi, F. Kubooka, N. Hamada, and N. Funabiki, "Group signature schemes with membership revocation for large groups," Proc. 10th Australasian Conference on Information Security andPrivacy (ACISP 2005), LNCS 3574, pp.443–454, Springer–Verlag, 2005.
24. T. Nakanishi and Y. Sugiyama, "A group signature scheme with efficient membership revocation for reasonable groups," Proc. 9th Australasian Conference on Information Security and Privacy (ACISP 2004), LNCS 3108, pp.336–347, Springer–Verlag, 2004.
25. I. Teranishi and K. Sako, "k-times anonymous authentication with a constant proving cost," Proc. 9th International Conference on Theory and Practice of Public-Key Cryptography (PKC 2006), LNCS 3958, pp.525–542, Springer–Verlag, 2006.