Electronic Preprint for Journal of Information Processing Vol.27

Regular Paper

Efficiency Improvement in Group Signature Scheme with Probabilistic Revocation

Nasima Begum[1,a),b)] Toru Nakanishi[2,c)]

Received: November 23, 2018, Accepted: June 11, 2019

Abstract: In group signature schemes, one of the important issues is the member revocation, and lots of revocable schemes have been proposed. Recently, Group Signature scheme with Probabilistic Revocation (GSPR) is proposed. In GSPR, by employing a novel notion of probabilistic revocation, the computation cost of the revocation check is drastically reduced, although the correctness of the check is with a certain probability. However, in the GSPR scheme, there is another problem: m alias tokens are embedded into the certificate of a member. Then, in signing, each token is used, and $O(m)$ exponentiations are needed to prove that the used token is embedded in the certificate. When m is large, this signing cost including $O(m)$ exponentiations becomes a big problem for powerless mobile devices. In this paper, we propose an extended GSPR scheme where the signing cost is reduced, but the revocation mechanism is exactly the same as the original GSPR scheme. Our main idea is to utilize an efficient pairing-based accumulator with multiplications to embed lots of alias tokens in a certificate. Thus, in the proposed scheme, the signing cost is reduced to only $O(m)$ multiplications instead of $O(m)$ exponentiations.

Keywords: anonymity, group signatures, revocation, accumulator

1. Introduction

1.1 Backgrounds

A group signature (GS) scheme [8] allows a group member to anonymously sign a message on behalf of the group. In a GS scheme, two types of trusted parties participate: A group manager (GM) has the authority to add a user to the own group, and an opener can identify the signer from a signature. The applications of GS include attribute-based anonymous credentials [7] and privacy-preserving cooperative intelligent transport systems [18]. One of the most important issues in GS is a revocation that the signing capability of a user is revoked. The revocation may happen, when the user leaves the group voluntarily or the account is banned due to the illegal usage, etc.

1.2 Previous Works

Lots of revocable GS (RGS) schemes have been proposed, which can be divided into two types. In the first type (e.g., Refs. [4], [13], [14], [17]), the signer needs the current revocation list (RL) of revocation tokens for signing. A revoked signer cannot compute a valid signature based on the RL. The recent RGS schemes in this type achieve the good asymptotic efficiency. The state-of-the-art scheme [14] achieves the constant signing/verification time, the constant size of signature and membership certificate, and $O(\log N)$ public key size, where N is the maximum number of group members. However, the RL size is $O(R)$, where R is the number of revoked members. Since the signer needs to fetch the RL for every revocation epoch, the large size will cause a delay in a mobile environment with lots of members.

The second type of RGS is the VLR (Verifier-Local Revocation)-GS [5], [16]. In VLR-GS, the revocation check based on the RL takes place at only the verifier, and thus the signer does not need to fetch the RL, which is suitable for mobile situations. In this type, each revocation token in the RL is generated from a member's secret key. The group signature includes a revocation-related part based on the secret key (Note that the computation of this part does not need the RL). The verifier can check whether a signer has been revoked or not, by a computation between the revocation-related part in the signature and each revocation token. However, the computational cost of the revocation check in each verification is $O(R)$ heavy pairings. Thus, as R increases, the large verification time is required.

In Ref. [12], as an improvement of VLR-GS, a GS scheme with probabilistic revocation (GSPR) has been proposed. GSPR significantly reduces the computation cost of the revocation check in the verification, using alias codes, instead of pairing relations. In GSPR, an alias code which is a vector of $+1$s and -1s is revealed in each group signature. The alias code is mapped from an alias token which is embedded in the certificate issued from the GM. When revoking a member, the alias codes from the alias tokens of the member are added to a united code RC. In the revocation check, the verifier checks that the alias code of the group signature is not added in RC using a single cross-correlation between the alias code and RC. Thus, the computation of the revocation check does not need any expensive cryptographic operation, and

[1] Department of Computer Science and Engineering, University of Asia Pacific, Bangladesh.
[2] Department of Information Engineering, Hiroshima University, Higashi-Hiroshima, Hiroshima 739–8527, Japan
[a)] mail4nasima@gmail.com
[b)] nasima.cse@uap-bd.edu
[c)] t-nakanishi@hiroshima-u.ac.jp

thus is very low. On the other hand, as the trade-off, the success of the revocation check is probabilistic, i.e., the correctness of its result is not ensured with certainty, but only with a certain probability.

In the GSPR scheme of Ref. [12], there is another weakness: m alias tokens are embedded into the certificate of a member, and each token is used in each unlinkable group signature. In signing, $O(m)$ exponentiations are needed. When all m alias tokens are used, a new certificate has to be re-issued from the GM. Thus, since m should be as large as possible, the signing cost including $O(m)$ exponentiation is a big problem for powerless mobile devices.

1.3 Our Contributions

In this paper [*1], we propose an extended GSPR scheme where the signing cost is reduced, but the revocation mechanism is exactly the same as the original GSPR scheme [12]. Our main idea is to utilize an efficient pairing-based accumulator [6] to embed alias tokens in a certificate. The accumulator and the corresponding witness are computed by only multiplications instead of exponentiations, and the verification of the accumulation is a single pairing relation. Thus, in the proposed scheme, signing needs only $O(m)$ multiplications instead of $O(m)$ exponentiations. Hence, we can reduce the signing cost. Since the same revocation mechanism using random alias tokens is adopted, the cost of the revocation check is still low.

However, in the proposed scheme, the verification cost has a slight constant overhead compared to the original. As another trade-off, the public key size is $O(N \cdot m)$. But, since the public key is communicated to each entity only one time, we consider that this is not so serious problem.

1.4 Related Works

As the related works, efficient revocable signature schemes have been proposed in Refs. [9], [20]. The characteristic of these schemes is to introduce the linkability to group signatures.

In Ref. [9], time intervals are introduced, and during each time interval, the group signatures are linkable, i.e., any verifier can determine whether the signer of a signature is the same as that of another signature. On the other hand, signatures in different intervals are unlinkable. Based on this characteristic, the scheme is applied to a road-to-vehicle communication system in Ref. [9]. The advantage of the scheme is the efficient revocation check. In the revocation, a linkable part in the signature is added to the revocation list (RL), and the verifier checks whether the part is in the list or not. This check can be efficiently executed by a hash table with no cryptographic operations. The advantage over our GSPR scheme is the signing cost. The linkable scheme achieves $O(1)$ signing cost, while our GSPR scheme need $O(m)$ multiplications. Thus, the scheme of Ref. [9] is suitable for the applications where the linkability in an interval with some moderate length of time is useful. On the other hand, from the viewpoint of privacy, to reduce the number of linkable signatures, a short interval is desired. However, in the scheme of Ref. [9], the time indicating the interval has to be synchronized, i.e., the signer has to know the current time authenticated by a server, and the verifier has to fetch the RL of the corresponding time. Furthermore, the whole of the RL dependent to the time has to be re-computed in the beginning of each interval, and the computation cost is $O(R)$ exponentiations. Thus, in case of the short interval and lots of revocations, the cost may be the problem. In our GSPR scheme, the signer does not need to know the time, and the computation of RL is very light.

In the scheme of Ref. [20], a TTP called Revocation Authority (RA) is introduced for efficient revocation check, as follows. To the group signature, an encrypted linkable part is attached. For the revocation check, the verifier sends the part to an online RA, and the RA decrypts the part, and checks the revocation by matching it to the part in RL in the same way as Ref. [9]. The advantage of this scheme is that the verifier does not need to fetch the RL and the computation of the revocation check, in addition to no cost for the signer as in the GSPR schemes. However, the demerit is that the RA can link any pair of all signatures of all users. In Ref. [20], the distributed RAs are considered. But, this needs the distributed decryption by the distributed multiple RAs, which causes the delay of the revocation check.

1.5 Outline

The rest of this paper is organized as follows: The used cryptographic tools are explained in Section 2. The syntax and security model of GSPR are defined in Section 3. Then, in Section 4, our efficient GSPR scheme is proposed, and the security is proved. The efficiency of our proposed scheme is compared with the previous GSPR scheme [12] in Section 5. Finally, we conclude this paper in Section 6.

2. Preliminaries

In this section, we show the cryptographic tools and the proof system used as building blocks of the proposed GSPR scheme.

2.1 Bilinear Maps

Our scheme utilizes the following bilinear groups:

(1) $\mathcal{G}_1$, $\mathcal{G}_2$, and $\mathcal{T}$ are cyclic groups of prime order p,

(2) g_1 and g_2 are randomly chosen generators of $\mathcal{G}_1$ and $\mathcal{G}_2$, respectively,

(3) e is an efficiently computable bilinear map: $e : \mathcal{G}_1 \times \mathcal{G}_2 \to \mathcal{T}$, i.e., **(a)** for all $u \in \mathcal{G}_1$, $v \in \mathcal{G}_2$ and $a, b \in \mathcal{Z}$, $e(u^a, v^b) = e(u, v)^{ab}$, **(b)** $e(g_1, g_2) \neq 1_{\mathcal{T}}$, where $1_{\mathcal{T}}$ is the identity element of group $\mathcal{T}$.

The bilinear map e can be efficiently implemented with the pairings. There are two types of bilinear pairings, symmetric ($\mathcal{G}_1 = \mathcal{G}_2$) and asymmetric ($\mathcal{G}_1 \neq \mathcal{G}_2$). In the following descriptions, for simplicity, we adopt the symmetric one, and let $e : \mathcal{G} \times \mathcal{G} \to \mathcal{T}$.

Remark 1. *The asymmetric pairing can be more efficiently implemented than the symmetric pairing. Thus, for the efficiency in the implementation, the adoption of an asymmetric pairing is better. But, the descriptions of the construction and the assumptions may become more complex. Our main contribution is to reduce $O(m)$ exponentiations in signing of the GSPR scheme to $O(m)$ multiplications, and this can be achieved by adopting an accumulator and a structure-preserving signature. Thus, for*

[*1] The preliminary version of this paper was presented at ISITA2018 [3].

the simplicity of the description, we adopt the symmetric pairing hereafter.

2.2 Complexity Assumptions

The security of our scheme is based on the q-SFP (Simultaneous Flexible Pairing) assumption [1], [2] for the utilized structure-preserving signatures. We also adopt the n-DHE (DH Exponent) assumption [6] for the utilized accumulator.

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

$$\begin{aligned}&\Pr[\mathcal{A}(g_z, h_z, g_r, h_r, a, \tilde{a}, b, \tilde{b}, \{(z_j, r_j, s_j, t_j, u_j, v_j, w_j)\}_{j=1}^{q})\\ &= (z^*, r^*, s^*, t^*, u^*, v^*, w^*) \in \mathcal{G}^7\\ &\wedge e(a, \tilde{a}) = e(g_z, z^*)e(g_r, r^*)e(s^*, t^*)\\ &\wedge e(b, \tilde{b}) = e(h_z, z^*)e(h_r, u^*)e(v^*, w^*)\\ &\wedge z^* \neq 1_{\mathcal{G}} \wedge z^* \neq z_j \text{ for all } 1 \leq j \leq q]\end{aligned}$$

is negligible, where $(g_z, h_z, g_r, h_r, a, \tilde{a}, b, \tilde{b}) \in \mathcal{G}^8$ and all tuples $\{(z_j, r_j, s_j, t_j, u_j, v_j, w_j)\}_{j=1}^{q}$ satisfy

$$\begin{aligned}&e(a, \tilde{a}) = e(g_z, z_j)e(g_r, r_j)e(s_j, t_j)\\ &\wedge e(b, \tilde{b}) = e(h_z, z_j)e(h_r, u_j)e(v_j, w_j),\end{aligned}$$

and $1_{\mathcal{G}}$ is the identity element of group $\mathcal{G}$.

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

$$\Pr[\mathcal{A}(g, g^a, \ldots, g^{a^n}, g^{a^{n+2}}, \ldots, g^{a^{2n}}) = g^{a^{n+1}}]$$

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

2.3 Structure-preserving Signatures (AHO Signatures)

We utilize the structure-preserving signatures. In the structure-preserving signatures, the verification keys, messages, and signatures are elements of bilinear groups, and the verification predicate is a conjunction of pairing products. As in the group signatures of Ref. [14], we adopt the AHO signature scheme in Refs. [1], [2].

- **AHOKeyGen:** Select $g, G_r, H_r \in_R \mathcal{G}$, and $\mu_z, \nu_z, \mu, \nu, \alpha_a, \alpha_b \in_R Z_p$. Compute $G_z = G_r^{\mu_z}$, $H_z = H_r^{\nu_z}$, $G = G_r^{\mu}$, $H = H_r^{\nu}$, $A = e(G_r, g^{\alpha_a})$, $B = e(H_r, g^{\alpha_b})$. Output the public key as $pk = (g, G_r, H_r, G_z, H_z, G, H, A, B)$, and the secret key as $sk = (\alpha_a, \alpha_b, \mu_z, \nu_z, \mu, \nu)$.
- **AHOSign:** Given message $M \in \mathcal{G}$ to be signed together with sk, choose $\beta, \epsilon, \eta, \iota, \kappa \in_R Z_p$, and compute $\theta_1 = g^{\beta}$, and

$$\begin{aligned}&\theta_2 = g^{\epsilon - \mu_z\beta}M^{-\mu}, \quad \theta_3 = G_r^{\eta}, \quad \theta_4 = g^{(\alpha_a - \epsilon)/\eta},\\ &\theta_5 = g^{\iota - \nu_z\beta}M^{-\nu}, \quad \theta_6 = H_r^{\kappa}, \quad \theta_7 = g^{(\alpha_b - \iota)/\kappa}.\end{aligned}$$

  Output the signature $\sigma = (\theta_1, \ldots, \theta_7)$.
- **AHOVerify:** Given the message M and the signature $\sigma = (\theta_1, \ldots, \theta_7)$, accept these if the following equations hold:

$$\begin{aligned}&A = e(G_z, \theta_1) \cdot e(G_r, \theta_2) \cdot e(\theta_3, \theta_4) \cdot e(G, M),\\ &B = e(H_z, \theta_1) \cdot e(H_r, \theta_5) \cdot e(\theta_6, \theta_7) \cdot e(H, M).\end{aligned}$$

Under the q-SFP assumption, this signature is existentially unforgeable against the chosen-message attack [1], [2]. Using the re-randomization algorithm in Refs. [1], [2], this signature can be publicly randomized to obtain another signature $(\theta'_1, \ldots, \theta'_7)$ on the same message. As a result, in the zero-knowledge proof, $(\theta'_i)_{i=3,4,6,7}$ can be safely revealed, while $(\theta'_i)_{i=1,2,5}$ have to be committed, as mentioned in Ref. [14].

2.4 Pairing-based Accumulator with Multiplications

The cryptographic accumulator transforms a large set of values into a single value, where the membership can be verified. In Ref. [6], an efficient pairing-based accumulator is proposed. The accumulator is computed using multiplications, instead of exponentiations used in other accumulator schemes. It can be verified using a single pairing relation with the constant complexity. In this accumulator, integer elements from the universal set $\{1, \ldots, n\}$ are accumulated. We define the syntax as follows.

- **AccSetup:** This is the algorithm to output the public parameters pk_{acc} and it is executed only once.
- **AccGen:** Given pk_{acc} and a set $V \subset \{1, \ldots, n\}$, this algorithm computes the accumulator acc_V of V.
- **AccWitGen:** Given pk_{acc}, $V \subset \{1, \ldots, n\}$ and $i \in V$, this algorithm computes the witness W for $i \in V$.
- **AccVerify:** Given pk_{acc}, acc_V, $i \in V$ and W, this algorithm verifies $i \in V$.

The correctness and the security of the accumulator are as follows:

Correctness: The accumulator is *correct* if, when **AccSetup** correctly computes pk_{acc} and **AccGen** and **AccWitGen** correctly output acc_V of V and W for $i \in V$, then **AccVerify** accepts them.

Security: Let us consider the following game between a challenger and the adversary:

Game$_{Acc}$:

(1) The challenger runs **AccSetup** to generate the public parameters pk_{acc}. It gives pk_{acc} to the adversary.

(2) The adversary outputs $V \subset \{1, \ldots, n\}$, $i \in \{1, \ldots, n\}$, and W. The challenger runs **AccGen** to generate the correct accumulator acc_V of V. Then the adversary wins if
   - **AccVerify** accepts pk_{acc}, acc_V, i, and W, but
   - $i \notin V$.

Then, the accumulator is *secure* if any PPT adversary can win **Game**$_{Acc}$ only with negligible probability.

The construction of the pairing-based accumulator with multiplications in Ref. [6] is as follows.

- **AccSetup:** Select $g \in_R \mathcal{G}$. Select a random value $\gamma \in_R Z_p$, and compute and output $pk_{acc} = (g, g_1 = g^{\gamma^1}, \ldots, g_n = g^{\gamma^n}, g_{n+2} = g^{\gamma^{n+2}}, \ldots, g_{2n} = g^{\gamma^{2n}}, z = e(g, g)^{\gamma^{n+1}})$ as the public parameters.
- **AccGen:** Given pk_{acc} and $V \subset \{1, \ldots, n\}$, compute and output $acc_V = \prod_{j \in V} g_{n+1-j}$.
- **AccWitGen:** Given pk_{acc}, $V \subset \{1, \ldots, n\}$ and $i \in V$, compute and output the witness as $W = \prod_{j \in V}^{j \neq i} g_{n+1-j+i}$.
- **AccVerify:** Given pk_{acc}, acc_V, $i \in V$ and W, accept if $e(g_i, acc_V)/e(g, W) = z$.

The security of the accumulator in this construction is satisfied under the n-DHE assumption [6].

2.5 Signature Proof of Knowledge (SPK)

As in the original GSPR [12], we use the Signature Proof of Knowledge (SPK) as the non-interactive zero-knowledge-proof technique. The SPK is transformed from a zero-knowledge proof of knowledge (PK), Σ-*protocol* on discrete logs via Fiat-Shamir transformation.

In this paper, the following SPKs on $\mathcal{G}$ and $\mathcal{T}$ 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 also be constructed on group $\mathcal{T}$.

SPK of representations with equal parts: An SPK proving the knowledge of representations of $C, C' \in \mathcal{G}$ to the bases $g_1, g_2, \ldots, g_t \in \mathcal{G}$ 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 $\mathcal{G}$ and $\mathcal{T}$ 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.

3. Syntax and Security Model

The algorithms of the Group Signature Probabilistic Revocation (GSPR) are as follows [12].

- **KeyGen:** This algorithm outputs a group public key gpk and GM's secret key gms.
- **Join:** Given gpk, gms, and member's ID $i \in [1, N]$ for the maximum number of members N, this algorithm outputs the member's secret key $\boldsymbol{gsk}_i$, revocation token $\boldsymbol{grt}_i$, and the i-th registration entry $\boldsymbol{reg}_i$. Here, we use the notation $\boldsymbol{gsk}$ as the list $\boldsymbol{gsk} = (\boldsymbol{gsk}_1, \boldsymbol{gsk}_2, \ldots, \boldsymbol{gsk}_N)$, and this is similar for $\boldsymbol{grt}$ and $\boldsymbol{reg}$.
- **Sign:** Given gpk, $\boldsymbol{gsk}_i$, and message M, this algorithm generates the group signature σ on M by member i.
- **Verify:** Given gpk, message M and the signature σ, this algorithm outputs valid if the signature is valid and not revoked. Otherwise, it outputs invalid. This algorithm consists of two sub-algorithms: **SignCheck** checks that the signature is correctly computed by a member. **RevCheck** checks that the signer is not revoked.
- **Revoke:** Given a revocation token $\boldsymbol{grt}_i$, this algorithm updates the revocation code RC, which is the revocation list in GSPR, s.t. member i is revoked.
- **Open:** Given $\boldsymbol{reg}$, signature σ, and message M, this algorithm outputs the signer's ID i.

Then, the security requirements are as follows.

Definition 3 (**Signature Correctness**). *For all* (gpk, gms) *obtained by* **KeyGen**, *all* $(\boldsymbol{gsk}_i, \boldsymbol{grt}_i, \boldsymbol{reg}_i)$ *obtained by* **Join** *for any* $i \in [1, N]$, *and all* $M \in \{0,1\}^*$,

$$\textbf{SignCheck}(gpk, \textbf{Sign}(gpk, \boldsymbol{gsk}_i, M), M) = \mathsf{valid}.$$

Definition 4 (**Identity Correctness**). *For all* (gpk, gms) *obtained by* **KeyGen**, *all* $(\boldsymbol{gsk}_i, \boldsymbol{grt}_i, \boldsymbol{reg}_i)$ *obtained by* **Join** *for any* $i \in [1, N]$, *and all* $M \in \{0,1\}^*$,

$$\textbf{Open}(\boldsymbol{reg}, \textbf{Sign}(gpk, \boldsymbol{gsk}_i, M), M) = i.$$

Definition 5 (**Revocation Correctness**). *For all* (gpk, gms) *obtained by* **KeyGen**, *all* $(\boldsymbol{gsk}_i, \boldsymbol{grt}_i, \boldsymbol{reg}_i)$ *obtained by* **Join** *for any* $i \in [1, N]$, *all* $M \in \{0,1\}^*$, *and all RC for which* **Revoke**$(\boldsymbol{grt}_i)$ *has never been invoked (i.e., member i is not revoked),*

$$\textbf{RevCheck}(RC, \textbf{Sign}(gpk, \boldsymbol{gsk}_i, M)) = \mathsf{valid}.$$

Definition 6 (**Anonymity**). *For any PPT algorithm* $\mathcal{A}$, *the advantage of* $\mathcal{A}$ *on winning the following game is negligibly small.*

- *Setup: The challenger runs* **KeyGen** *and* **Join** (gpk, gms, i) *for* $\forall i \in [1, n]$. *He obtains gpk,* $\boldsymbol{gsk}$ *and* $\boldsymbol{reg}$. *He runs* $\mathcal{A}$ *with gpk.*
- *Queries-Phase I:* $\mathcal{A}$ *queries the challenger about the following:*
  - *Signing:* $\mathcal{A}$ *requests a signature on an arbitrary message M for an arbitrary member i. The challenger responds with the corresponding signature.*
  - *Corruption:* $\mathcal{A}$ *requests the secret key of an arbitrary member i. The challenger responds with the key* $\boldsymbol{gsk}_i$.
  - *Opening:* $\mathcal{A}$ *requests the identity of the signer by providing a message M and its valid signature* σ *created by signer* $i \in [1, n]$. *The challenger responds with the signer's identity i.*
- *Challenge:* $\mathcal{A}$ *outputs a message* M^* *and two members* i_0 *and* i_1, *where the corruption of* i_0 *and* i_1 *have not been requested. The challenger chooses* $\phi \leftarrow \{0,1\}$, *and responds with the signature* σ^* *on* M^* *of member* i_ϕ.
- *Queries-Phase II (Restricted Queries): After the challenge,* $\mathcal{A}$ *can make additional queries of signing, corruption and opening, except the corruption queries of* i_0 *and* i_1, *and the opening query of the signature* σ^* *on* M^*.
- *Output:* $\mathcal{A}$ *outputs a bit* ϕ' *indicating its guess of* ϕ. $\mathcal{A}$ *wins the anonymity game if* $\phi' = \phi$. *The advantage of* $\mathcal{A}$ *is defined as* $|Pr(\phi' = \phi) - 1/2|$.

Remark 2. *In the previous paper [12], the authors consider the extended situation, where time intervals are introduced, and in the same interval, a user uses the same alias token to generate the group signature. However, in the situation, the signatures in the same interval are linkable w.r.t. the sameness of the signer. In such a situation, the definition of anonymity should consider the linkability. But, in the previous paper, only the basic anonymity where the linkability is not considered is defined. In this paper, as the main contributions, we target the efficiency improvement of*

signing while the security remains. Thus, for simplicity, we adopt this definition of basic anonymity without the linkability, and do not consider the extended situation where the same alias token is used in the same interval.

Definition 7 (**Traceability**). *For each PPT algorithm $\mathcal{A}$, the probability that $\mathcal{A}$ wins the following game is negligibly small.*

- *Setup: The challenger runs* **KeyGen** *and* **Join** (gpk, gms, i), $\forall i \in [1, n]$. *He obtains gpk,* $\boldsymbol{gsk}$ *and* $\boldsymbol{reg}$. *He provides the adversary $\mathcal{A}$ with gpk, and sets U as empty set.*
- *Queries: $\mathcal{A}$ queries the challenger about the following:*
  - *Signing: $\mathcal{A}$ requests a signature on an arbitrary message M for an arbitrary member i. The challenger returns the corresponding signature σ to $\mathcal{A}$.*
  - *Corruption: $\mathcal{A}$ requests the secret key of an arbitary member i. The challenger appends i to U, and responds with the key $\boldsymbol{gsk}_i$.*
- *Output: $\mathcal{A}$ outputs a message M^*, a set RL^* of revoked member ID's and a signature σ^*.*
  The forgery $\mathcal{A}$ wins the game if
  (1) σ^* is accepted by the verification algorithm as a valid signature on M^* for the revoked members RL^*, i.e, **SignCheck**$(gpk, \sigma^*, M^*) = \mathsf{valid}$, *and* **RevCheck**$(RC, \sigma^*) = \mathsf{valid}$, *where RC is generated by invoking* **Revoke**$(\boldsymbol{grt}_i)$ *for all $i \in RL^*$.*
  (2) σ^* is traced by **Open** *to a member outside of the coalition $U \backslash RL^*$ (i.e., the traced member is not corrupted, or the traced member is revoked) or* **Open** *algorithm fails, and*
  (3) σ^* is nontrivial, i.e., $\mathcal{A}$ did not obtain σ^* by making a signing query on M^*.

4. Proposed Efficient GSPR Scheme

4.1 Construction Idea

Since our aim is to improve the signing costs in the GSPR scheme [12], we first review the previous GSPR scheme. In the scheme, for efficient revocation check, alias codes are introduced. Each alias code is a vector of +1s and −1s. The alias code is mapped from an alias token of a random integer. Alias tokens are embedded into the certificate of $\boldsymbol{gsk}_i$. In **Sign**, an unused alias token is revealed as a tag for revocation. In **Revoke**, alias codes are mapped from the alias tokens of the revoked member, and the codes are added to *RC*. **RevCheck** checks that the alias code of the alias token revealed in the signature is not added to *RC* using a single cross-correlation of the alias code and *RC*. Thus, the computation of **RevCheck** is very low. As the trade-off, the success of **RevCheck** is probabilistic, i.e., the correctness of its result is not ensured with certainty, but only with a certain probability.

However, for the verification that a used alias token is embedded in the certificate, a witness is computed in **Sign**. The witness computation includes $O(m)$ exponentiations, where m is the number of embedded alias tokens. When all of m alias tokens are used, a new $\boldsymbol{gsk}_i$ has to be re-issued. Thus, since m should be as large as possible, the witness computation cost can be a burden for mobile devices with small computation power.

Our main idea for efficiency improvement is to adopt the efficient accumulator with multiplications for embedding alias tokens into the certificate. In this accumulator, the witness computation needs only $O(m)$ multiplications instead of exponentiations. Hence, the computation cost in **Sign** is reduced. On the other hand, the public key size increases, since the size depends on m and N.

The accumulator of the alias tokens is generated as a group element of $\mathcal{G}$. The accumulator has to be signed as the certificate of the accumulator, and the verification has to be proved by a zero-knowledge proof. Thus, in our proposed scheme, we utilize the AHO signature to sign the accumulator of alias tokens as the certificate. In the group signature, the verifications of the AHO signature and the accumulator including the used alias token are proved by SPKs for anonymity.

Remark 3. *In our construction, as in the group signature schemes in Ref. [14], the AHO signature scheme is adopted as the structure-preserving signature (SPS) scheme. After the AHO signature scheme, efficient SPS schemes based on simple assumptions (non-q-type assumptions) have been proposed in Refs. [11], [15]. By adapting these schemes to our approach, we may obtain the GSPR scheme with more efficiency and/or simpler assumptions. Our contribution of this work is to reduce $O(m)$ exponentiations in the signing cost to $O(m)$ multiplications, which is achieved by adopting the accumulator [6] and an SPS scheme including the AHO signature scheme.*

4.2 Proposed Construction

The following are the details of our proposed construction. In this proposed scheme, random integers from 1 to n are used as alias tokens (In the original scheme [12], outputs of a hash function from a member's secret are used). Each of N group members is assigned to m random integers. The integers of a member cannot be overlapped with other integers. Thus, $N \cdot m$ integers are needed in total, and let $n = N \cdot m$.

- **KeyGen:** Select bilinear groups $\mathcal{G}$, $\mathcal{T}$ with a prime order p and a bilinear map e. Using **AHOKeyGen**, generate the public key pk_{AHO} and secret key sk_{AHO}. Using **AccSetup**, generate the public parameters for accumulator pk_{acc} for the universal set $\{1, \ldots, n\}$. For commitments, choose $\hat{g} \in_R \mathcal{G}$. Then, output $gpk = (\mathcal{G}, \mathcal{T}, p, e, pk_{\mathrm{AHO}}, pk_{acc}, \hat{g})$ and $gms = sk_{\mathrm{AHO}}$.
- **Join**(gpk, gms, i)**:** Let V_i be the set of alias tokens (random integers) for member i, where $|V_i| = m$ (Note that $V_i \cap V_j = \emptyset$ for V_j of any other member j).
  (1) Compute the accumulator acc_{V_i} for set V_i as:
  $$acc_{V_i} = \prod_{j \in V_i} g_{n+1-j}.$$
  (2) Compute an AHO signature for the acc_{V_i} using sk_{AHO}. The signature is denoted as $\tilde{\sigma}_{acc_{V_i}} = (\tilde{\theta}_1, \ldots, \tilde{\theta}_7)$. As the outputs of this algorithm, the secret key is $\boldsymbol{gsk}_i = (V_i, acc_{V_i}, \tilde{\sigma}_{acc_{V_i}})$, the revocation token is $\boldsymbol{grt}_i = V_i$, and the registration entry is $\boldsymbol{reg}_i = V_i$.
- **Sign**$(gpk, \boldsymbol{gsk}_i, M)$**:**
  (1) Select one of an unused alias token $v_{i,k} \in V_i$. Then compute the witness of $v_{i,k} \in V_i$ as:

$$W = \prod_{j \in V_i}^{j \neq v_{i,k}} g_{n+1-j+v_{i,k}}.$$

(2) Re-randomize the AHO signature $\tilde{\sigma}_{acc_{V_i}}$ to obtain $\tilde{\sigma}'_{acc_{V_i}} = (\tilde{\theta}'_1, \ldots, \tilde{\theta}'_7)$, using the method in Ref. [2], which needs 8 exponentiations on $\mathcal{G}$-element.

(3) Select random values $r_{acc_{V_i}} \in_R Z_p$ and $r_W \in_R Z_p$. Compute the commitment $C_{acc_{V_i}}$ to acc_{V_i} by $C_{acc_{V_i}} = acc_{V_i}\hat{g}^{r_{acc_{V_i}}}$, and the commitment C_W to W by $C_W = W\hat{g}^{r_W}$.

(4) Select random values $r_{\tilde{\theta}'_i} \in_R Z_p$ for $i \in \{1, 2, 5\}$. Compute the commitments $\{C_{\tilde{\theta}'_i}\}_{i\in\{1,2,5\}}$ to $\{\tilde{\theta}_{i'}\}_{i\in\{1,2,5\}}$ by $C_{\theta'_i} = \tilde{\theta}'_i\hat{g}^{r_{\tilde{\theta}'_i}}$. Set $\boldsymbol{com}_{\mathrm{AHO}} = (\{\tilde{\theta}_{i'}\}_{i=3,4,6,7}, \{C_{\tilde{\theta}'_i}\}_{i=1,2,5})$.

(5) Generate the following SPK π on message M:

$$\begin{aligned}
&SPK\{(r_{\tilde{\theta}'_1}, r_{\tilde{\theta}'_2}, r_{\tilde{\theta}'_5}, r_{acc_{V_i}}, r_W) :\\
&A^{-1} \cdot e(G_z, C_{\tilde{\theta}'_1}) \cdot e(G_r, C_{\tilde{\theta}'_2}) \cdot e(\tilde{\theta}'_3, \tilde{\theta}'_4) \cdot e(G, C_{acc_{V_i}})\\
&\quad = e(G_z, \hat{g})^{r_{\tilde{\theta}'_1}} \cdot e(G_r, \hat{g})^{r_{\tilde{\theta}'_2}} \cdot e(G, \hat{g})^{r_{acc_{V_i}}}, \qquad (1)\\
&B^{-1} \cdot e(H_z, C_{\tilde{\theta}'_1}) \cdot e(H_r, C_{\tilde{\theta}'_5}) \cdot e(\tilde{\theta}'_6, \tilde{\theta}'_7) \cdot e(H, C_{acc_{V_i}})\\
&\quad = e(H_z, \hat{g})^{r_{\tilde{\theta}'_1}} \cdot e(H_r, \hat{g})^{r_{\tilde{\theta}'_5}} \cdot e(H, \hat{g})^{r_{acc_{V_i}}}, \qquad (2)\\
&e(g_{v_{i,k}}, C_{acc_{V_i}}) \cdot e(g, C_W)^{-1} \cdot z^{-1}\\
&\quad = e(g_{v_{i,k}}, \hat{g})^{r_{acc_{V_i}}} \cdot e(g, \hat{g})^{-r_W}\}(M) \qquad (3)
\end{aligned}$$

Equations (1), (2) show the verification of the AHO signature $\tilde{\sigma}'_{acc_{V_i}}$ on acc_{V_i}. Equation (3) shows the accumulator verification of $v_{i,k} \in V_i$.

This SPK π is computed with the Fiat-Shamir heuristic [10] by the following steps:

(a) Select randoms $\delta_{\tilde{\theta}'_1}$, $\delta_{\tilde{\theta}'_2}$, $\delta_{\tilde{\theta}'_5}$, $\delta_{acc_{V_i}}$, $\delta_W \in_R Z_p$ and compute:

$$\begin{aligned}
R_1 &= e(G_z, \hat{g})^{\delta_{\tilde{\theta}'_1}} \cdot e(G_r, \hat{g})^{\delta_{\tilde{\theta}'_2}} \cdot e(G, \hat{g})^{\delta_{acc_{V_i}}},\\
R_2 &= e(H_z, \hat{g})^{\delta_{\tilde{\theta}'_1}} \cdot e(H_r, \hat{g})^{\delta_{\tilde{\theta}'_5}} \cdot e(H, \hat{g})^{\delta_{acc_{V_i}}},\\
R_3 &= e(g_{v_{i,k}}, \hat{g})^{\delta_{acc_{V_i}}} \cdot e(g, \hat{g})^{-\delta_W}.
\end{aligned}$$

(b) Using a collision resistant hash function $\mathcal{H}$ treated as random oracles, compute the challenge c as:
$c = \mathcal{H}(gpk, M, v_{i,k}, C_{acc_{V_i}}, C_W, \boldsymbol{com}_{\mathrm{AHO}}, R_1, R_2, R_3)$.

(c) Compute the responses as:

$$\begin{aligned}
s_{\tilde{\theta}'_1} &= \delta_{\tilde{\theta}'_1} + c \cdot r_{\tilde{\theta}'_1},\\
s_{\tilde{\theta}'_2} &= \delta_{\tilde{\theta}'_2} + c \cdot r_{\tilde{\theta}'_2},\\
s_{\tilde{\theta}'_5} &= \delta_{\tilde{\theta}'_5} + c \cdot r_{\tilde{\theta}'_5},\\
s_{acc_{V_i}} &= \delta_{acc_{V_i}} + c \cdot r_{acc_{V_i}},\\
s_W &= \delta_W + c \cdot r_W.
\end{aligned}$$

Set $\pi = (c, s_{\tilde{\theta}'_1}, s_{\tilde{\theta}'_2}, s_{\tilde{\theta}'_5}, s_{acc_{V_i}}, s_W)$.

Note that the computation cost of this SPK depends on the number of the exponentiations by the knowledges $r_{\tilde{\theta}'_1}, r_{\tilde{\theta}'_2}, r_{\tilde{\theta}'_5}, r_{acc_{V_i}}, r_W$ in the proved equations (1)–(3).

(6) The output of this algorithm is the signature:
$\sigma = (v_{i,k}, C_{acc_{V_i}}, C_W, \boldsymbol{com}_{\mathrm{AHO}}, \pi)$.

- **Verify**(gpk, RC, σ, M)**:** Using the following sub-algorithms, if both of the sub-algorithms output valid, this algorithm outputs valid, otherwise outputs invalid.
  - **SignCheck**(gpk, σ, M)**:** Verify the SPK π in σ. If it is valid, this sub-algorithm outputs valid. Otherwise, it outputs invalid. This is checked using the following steps:

    (1) Retrieve:

$$\begin{aligned}
\tilde{R}_1 &= e(G_z, \hat{g})^{s_{\tilde{\theta}'_1}} \cdot e(G_r, \hat{g})^{s_{\tilde{\theta}'_2}} \cdot e(G, \hat{g})^{s_{acc_{V_i}}}\\
&\quad \cdot \{A^{-1} \cdot e(G_z, C_{\tilde{\theta}'_1}) \cdot e(G_r, C_{\tilde{\theta}'_2})\\
&\quad \cdot e(\tilde{\theta}'_3, \tilde{\theta}'_4) \cdot e(G, C_{acc_{V_i}})\}^{-c},\\
\tilde{R}_2 &= e(H_z, \hat{g})^{s_{\tilde{\theta}'_1}} \cdot e(H_r, \hat{g})^{s_{\tilde{\theta}'_5}} \cdot e(H, \hat{g})^{s_{acc_{V_i}}}\\
&\quad \cdot \{B^{-1} \cdot e(H_z, C_{\tilde{\theta}'_1}) \cdot e(H_r, C_{\tilde{\theta}'_5})\\
&\quad \cdot e(\tilde{\theta}'_6, \tilde{\theta}'_7) \cdot e(H, C_{acc_{V_i}})\}^{-c},\\
\tilde{R}_3 &= e(g_{v_{i,k}}, \hat{g})^{s_{acc_{V_i}}} \cdot e(g, \hat{g})^{-s_W}\\
&\quad \cdot \{e(g_{v_{i,k}}, C_{acc_{V_i}}) \cdot e(g, C_W)^{-1} \cdot z^{-1}\}^{-c}
\end{aligned}$$

    (2) Check the correctness of the challenge c as:
    $c = \mathcal{H}(gpk, M, v_{i,k}, C_{acc_{V_i}}, C_W, \boldsymbol{com}_{\mathrm{AHO}}, \tilde{R}_1, \tilde{R}_2, \tilde{R}_3)$.
    If the above equation holds, this sub-algorithm outputs valid; otherwise, it outputs invalid.
  - **RevCheck**(RC, σ)**:** For checking whether the alias token $v_{i,k}$ in σ has been revoked or not, do the following steps, which are the same as the original GSPR [12].
    (1) Map $v_{i,k}$ to the corresponding alias code $s_{i,k}$, i.e., compute $s_{i,k} = F_c(v_{i,k})$, where F_c is the map [12] from alias tokens to alias codes, and $s_{i,k}$ is a column vector of length l of samples of $+1$s and -1s.
    (2) Compute the value of the decision variable, $z = \frac{1}{l} {s_{i,k}}^T RC$, where ${s_{i,k}}^T$ is the transpose of $s_{i,k}$.
    (3) Output invalid if $z \geq \tau$, where τ is a pre-determined threshold, otherwise, output valid.
- **Revoke**$(\boldsymbol{grt}_i, RC)$**:** To revoke member i, update the revocation code RC using the following steps, which are the same as the original GSPR. In the original GSPR scheme, as the alias codes, piecewise-orthogonal codes are used. The alias codes are generated by concatenating multiple segments of orthogonal codes, and each alias code is represented by a vector of length l of $+1$s and -1s. RC is the summation of all alias codes of revoked users. In advance, the RC is initialized as the vector of length l of all 0s.
  (1) Map each $v_{i,k} \in \boldsymbol{grt}_i$ to the corresponding alias code $s_{i,k}$, i.e., compute $s_{i,k} = F_c(v_{i,k})$ for $k = 1, 2, \ldots, m$.
  (2) Compute the code, $\bar{s}_i$, by adding all the unique alias codes corresponding to the revoked alias tokens such that $\bar{s}_i = \sum_{k=1}^{m} s_{i,k}$.
  (3) Update the revocation code as $RC = RC + \bar{s}_i$.
- **Open**$(\boldsymbol{reg}, \sigma, M)$**:** The actual signer is identified using the following steps, which are the same as the original GSPR.
  (1) Search the registration list $\boldsymbol{reg}$ to find signer i that has generated signature σ with the alias token $v_{i,k}$.
  (2) If a match is successfully found, output i. Otherwise, output 0 to indicate a failure.

4.3 Security Discussions

Correctness. The signature and identity correctness is simply shown as in the original [12]. Since the revocation check mechanism is the same, the revocation correctness is the same as the original, where the revocation check for the non-revoked mem-

bers fails (i.e., **RevCheck** outputs invalid) with some probability, depending on the adopted code (For the details, see Ref. [12]).

Anonymity. We can prove the anonymity of our construction, as follows.

Theorem 1. *The proposed scheme satisfies the anonymity in the random oracle model.*

Proof. For the proposed scheme, consider **Game 0** and **Game 1**. **Game 0** is the original anonymity game between an adversary $\mathcal{A}$ and a challenger, where the hash function in SPKs are computed by the random oracle. **Game 0** is transformed to **Game 1** with the following changes:

- In *Challenge* phase, the signature $\sigma^* = (v^*_{i,k}, C^*_{acc_{V_i}}, C^*_W, \boldsymbol{com}^*_{\mathrm{AHO}}, \pi^*)$ with which the challenger responds is modified as follows. The commitments $C^*_{acc_{V_i}}, C^*_W$, and the re-randomized $\{\tilde{\theta}^*_{i'}\}_{i=3,4,6,7}$ and the commitments $\{C^*_{\tilde{\theta}'_i}\}_{i=1,2,5}$ in $\boldsymbol{com}^*_{\mathrm{AHO}}$ are modified to uniformly random elements from $\mathcal{G}$. The SPK π^* is modified to the zero-knowledge simulator, where the output of the hash function is backpacked by using the random oracle (In case of the failure of backpaching with some negligible probability, abort).

In **Game 1**, the random elements $C^*_{acc_{V_i}}, C^*_W, \boldsymbol{com}^*_{\mathrm{AHO}}$ and the zero-knowledge simulator π^* in the responded signature σ^* do not have any information on members i_0 and i_1. The remaining $v^*_{i,k}$ is a random integer and the assignment to the member (i.e., the revocation token $\boldsymbol{grt}_i = V_i$) is hidden to $\mathcal{A}$, and thus it does not have any information on i_ϕ. Therefore, the advantage of $\mathcal{A}$ in **Game 1**, i.e., the difference between the probability that $\mathcal{A}$ wins and 1/2 is only the probability of the failure of backpaching, which is negligible.

We can view **Game 0** and **Game 1** as operating in the same probability space, due to the perfectly hiding of the commitments and the re-randomized AHO signature and the perfect zero-knowledge-ness of SPK, except the negligible probability due to the backpaching failure. Thus, in **Game 0**, the advantage of $\mathcal{A}$ is also negligible. □

Traceability. We can prove the traceability of our construction, as follows.

Theorem 2. *The proposed scheme satisfies the traceability under the security of the AHO signatures and the accumulator.*

Proof. To win the traceability game, the adversary $\mathcal{A}$ must output a message M^*, a set RL^* of revoked members and a signature σ^* satisfying

(1) **Verify**$(gpk, \sigma^*, M^*, RL^*) =$ valid, and

(2) σ^* is traced to a member outside of the coalition $U \backslash RL^*$ or **Open** algorithm fails.

Let $\sigma^* = (v^*_{i,k}, C^*_{acc_{V_i}}, C^*_W, \boldsymbol{com}^*_{\mathrm{AHO}}, \pi^*)$, where $\boldsymbol{com}^*_{\mathrm{AHO}} = (\{\tilde{\theta}^*_{i'}\}_{i=3,4,6,7}, \{C^*_{\tilde{\theta}'_i}\}_{i=1,2,5})$. As mentioned in Section 2.5, from SPK π^*, we have an extractor the knowledge $r_{\tilde{\theta}'_1}$, $r_{\tilde{\theta}'_2}$, $r_{\tilde{\theta}'_5}$, $r_{acc_{V_i}}$, r_W satisfying Eqs. (1)–(3), as follows. This extractor is similar to the security proof of the SPK-based group signature scheme [4], which is constructed using the methodology of Forking Lemma in Ref. [19]. The extractor runs the traceability game with the adversary, and at first, the adversary as the prover in the underlying PK protocol of the SPK computes R_1, R_2, R_3 as the commitments, and requests the hash query on R_1, R_2, R_3. To the challenge c responded by the hash query, the prover outputs the responses $s_{\tilde{\theta}'_1}$, $s_{\tilde{\theta}'_2}$, $s_{\tilde{\theta}'_5}$, $s_{acc_{V_i}}$, s_W. Then, the extractor rewinds the prover to the point just before the prover is given challenge c. By the Forking Lemma, with non-negligible probability, for the same R_1, R_2, R_3 in the SPK, the prover is given challenge $c' \neq c$, the prover outputs the corresponding $s'_{\tilde{\theta}'_1}$, $s'_{\tilde{\theta}'_2}$, $s'_{\tilde{\theta}'_5}$, $s'_{acc_{V_i}}$, s'_W. Since the SPK is accepting, the following equations hold.

$$\begin{aligned} R_1 = {} & e(G_z, \hat{g})^{s_{\tilde{\theta}'_1}} \cdot e(G_r, \hat{g})^{s_{\tilde{\theta}'_2}} \cdot e(G, \hat{g})^{s_{acc_{V_i}}} \\ & \cdot \{A^{-1} \cdot e(G_z, C^*_{\tilde{\theta}'_1}) \cdot e(G_r, C^*_{\tilde{\theta}'_2}) \\ & \cdot e(\tilde{\theta}'_3{}^*, \tilde{\theta}'_4{}^*) \cdot e(G, C^*_{acc_{V_i}})\}^{-c}, \end{aligned} \tag{4}$$

$$\begin{aligned} R_2 = {} & e(H_z, \hat{g})^{s_{\tilde{\theta}'_1}} \cdot e(H_r, \hat{g})^{s_{\tilde{\theta}'_5}} \cdot e(H, \hat{g})^{s_{acc_{V_i}}} \\ & \cdot \{B^{-1} \cdot e(H_z, C^*_{\tilde{\theta}'_1}) \cdot e(H_r, C^*_{\tilde{\theta}'_5}) \\ & \cdot e(\tilde{\theta}'_6{}^*, \tilde{\theta}'_7{}^*) \cdot e(H, C^*_{acc_{V_i}})\}^{-c}, \end{aligned} \tag{5}$$

$$\begin{aligned} R_3 = {} & e(g_{v_{i,k}}, \hat{g})^{s_{acc_{V_i}}} \cdot e(g, \hat{g})^{-s_W} \\ & \cdot \{e(g_{v_{i,k}}, C^*_{acc_{V_i}}) \cdot e(g, C^*_W)^{-1} \cdot z^{-1}\}^{-c} \end{aligned} \tag{6}$$

For c', $s'_{\tilde{\theta}'_1}$, $s'_{\tilde{\theta}'_2}$, $s'_{\tilde{\theta}'_5}$, $s'_{acc_{V_i}}$, s'_W, the above equations also hold. Set $\Delta c = c - c'$, and $\Delta s_{\tilde{\theta}'_1} = s_{\tilde{\theta}'_1} - s'_{\tilde{\theta}'_1}$, and similarly for $s_{\tilde{\theta}'_2}, s_{\tilde{\theta}'_5}, s_{acc_{V_i}}, s_W$. For Eq. (4), we have

$$\begin{aligned} & e(G_z, \hat{g})^{s_{\tilde{\theta}'_1}} \cdot e(G_r, \hat{g})^{s_{\tilde{\theta}'_2}} \cdot e(G, \hat{g})^{s_{acc_{V_i}}} \\ & \quad \cdot \{A^{-1} \cdot e(G_z, C^*_{\tilde{\theta}'_1}) \cdot e(G_r, C^*_{\tilde{\theta}'_2}) \\ & \quad \cdot e(\tilde{\theta}'_3{}^*, \tilde{\theta}'_4{}^*) \cdot e(G, C^*_{acc_{V_i}})\}^{-c} \\ = {} & e(G_z, \hat{g})^{s'_{\tilde{\theta}'_1}} \cdot e(G_r, \hat{g})^{s'_{\tilde{\theta}'_2}} \cdot e(G, \hat{g})^{s'_{acc_{V_i}}} \\ & \quad \cdot \{A^{-1} \cdot e(G_z, C^*_{\tilde{\theta}'_1}) \cdot e(G_r, C^*_{\tilde{\theta}'_2}) \\ & \quad \cdot e(\tilde{\theta}'_3{}^*, \tilde{\theta}'_4{}^*) \cdot e(G, C^*_{acc_{V_i}})\}^{-c'}, \end{aligned}$$

and we obtain

$$\begin{aligned} & \{A^{-1} \cdot e(G_z, C^*_{\tilde{\theta}'_1}) \cdot e(G_r, C^*_{\tilde{\theta}'_2}) \cdot e(\tilde{\theta}'_3{}^*, \tilde{\theta}'_4{}^*) \cdot e(G, C^*_{acc_{V_i}})\}^{\Delta c} \\ & \quad = e(G_z, \hat{g})^{\Delta s_{\tilde{\theta}'_1}} \cdot e(G_r, \hat{g})^{\Delta s_{\tilde{\theta}'_2}} \cdot e(G, \hat{g})^{\Delta s_{acc_{V_i}}}. \end{aligned}$$

By taking Δc-th roots due to $\Delta c = c - c' \neq 0$, setting $r_{\tilde{\theta}'_1} = \Delta s_{\tilde{\theta}'_1} / \Delta c$, and similarly for $s_{\tilde{\theta}'_2}$, $s_{\tilde{\theta}'_5}$, $s_{acc_{V_i}}$, s_W, we can extract $r_{\tilde{\theta}'_1}$, $r_{\tilde{\theta}'_2}$, $r_{acc_{V_i}}$ s.t.

$$\begin{aligned} & A^{-1} \cdot e(G_z, C^*_{\tilde{\theta}'_1}) \cdot e(G_r, C^*_{\tilde{\theta}'_2}) \cdot e(\tilde{\theta}'_3{}^*, \tilde{\theta}'_4{}^*) \cdot e(G, C^*_{acc_{V_i}}) \\ & \quad = e(G_z, \hat{g})^{r_{\tilde{\theta}'_1}} \cdot e(G_r, \hat{g})^{r_{\tilde{\theta}'_2}} \cdot e(G, \hat{g})^{r_{acc_{V_i}}}, \end{aligned}$$

which is Eq. (1). For the other equations (5), (6), in the similar way, we can extract the knowledge $r_{\tilde{\theta}'_1}$, $r_{\tilde{\theta}'_5}$, $r_{acc_{V_i}}$, r_W satisfying Eqs. (2), (3).

From Eq. (1), we have the following equation:

$$\begin{aligned} A = {} & e(G_z, C^*_{\tilde{\theta}'_1} \hat{g}^{-r_{\tilde{\theta}'_1}}) \cdot e(G_r, C^*_{\tilde{\theta}'_2} \hat{g}^{-r_{\tilde{\theta}'_2}}) \cdot e(\tilde{\theta}'_3{}^*, \tilde{\theta}'_4{}^*) \\ & \cdot e(G, C^*_{acc_{V_i}} \hat{g}^{-r_{acc_{V_i}}}) \end{aligned}$$

Thus, we can extract $\theta'^*_1 = C^*_{\tilde{\theta}'_1} \hat{g}^{-r_{\tilde{\theta}'_1}}$, $\tilde{\theta}'_2{}^* = C^*_{\tilde{\theta}'_2} \hat{g}^{-r_{\tilde{\theta}'_2}}$, and $acc^*_{V_i} = C^*_{acc_{V_i}} \hat{g}^{-r_{acc_{V_i}}}$ s.t.

$$A = e(G_z, \tilde{\theta}'_1{}^*) \cdot e(G_r, \tilde{\theta}'_2{}^*) \cdot e(\tilde{\theta}'_3{}^*, \tilde{\theta}'_4{}^*) \cdot e(G, acc^*_{V_i}).$$

Similarly, from Eq. (2), we can extract $\tilde{\theta'_1}^*$, $\tilde{\theta'_5}^*$, and $acc^*_{V_i}$ s.t.

$$B = e(H_z, \tilde{\theta'_1}^*) \cdot e(H_r, \tilde{\theta'_5}^*) \cdot e(\tilde{\theta'_6}^*, \tilde{\theta'_7}^*) \cdot e(H, acc^*_{V_i}).$$

Furthermore, from Eq. (3), we have

$$e(g_{v^*_{i,k}}, C^*_{acc_{V_i}} \hat{g}^{-r_{acc_{V_i}}}) / e(g, C^*_W \hat{g}^{-r_W}) = z.$$

Thus, we can extract $acc^*_{V_i} = C^*_{acc_{V_i}} \hat{g}^{-r_{acc_{V_i}}}$ and $W^* = C^*_W \hat{g}^{-r_W}$ s.t.

$$e(g_{v^*_{i,k}}, acc^*_{V_i}) / e(g, W^*) = z.$$

Therefore, we can extract $acc^*_{V_i}$, W^* satisfying the accumulator verification with $acc^*_{V_i}$, and the re-randomized AHO signature $\{\theta'^*_1, \ldots, \theta'^*_7\}$ for $acc^*_{V_i}$.

We distinguish the following cases from each other.

- **Type 1 forgery.** The AHO signature on $acc^*_{V_i}$ was never issued.
- **Type 2 forgery.** The AHO signature on $acc^*_{V_i}$ was issued to a corrupted user. In this case, the signer has to be revoked from the following reason: the winning condition (2) of the traceability game means the signer has to be outside of U/RL^* (U means the corrupted users). However, in this case, this signer is issued the AHO signature by the corruption query, and so the signer is a corrupted user. This is why the signer has to be revoked.

Using Type 1 and Type 2 forgeries, we can obtain forgers against the AHO signatures and the accumulator respectively, as follows.

- **Type 1 Forgery.** With the adversary $\mathcal{A}$, simulate the traceability game, as follows. The public key of AHO signatures is given to the AHO signature forger. This is set as pk_{AHO} in **KeyGen**. Choose and compute the other parameters as in the real algorithms of **KeyGen**, and V_i in **Join** for each member i. Then, run $\mathcal{A}$ on gpk. In the traceability game, $\mathcal{A}$ can request corruption query for member i. Then, compute

  $$acc_{V_i} = \prod_{j \in V_i} g_{n+1-j}$$

  for V_i, and request the AHO signature on acc_{V_i} to the signing oracle of the AHO signatures. Then, respond $\boldsymbol{gsk}_i = (V_i, acc_{V_i}, \tilde{\sigma}_{acc_{V_i}})$ including the AHO signature on acc_{V_i}. The signing oracle in the traceability game can be simulated using the simulated commitments and the zero-knowledge simulator of SPK without $\boldsymbol{gsk}_i$. Finally, $\mathcal{A}$ outputs a forged group signature σ^*. As mentioned above, we can extract a randomized AHO signature on $acc^*_{V_i}$. In this case, since the AHO signature was never issued for $acc^*_{V_i}$, this implies the forgery against the AHO signature.
- **Type 2 Forgery.** This is the forgery against the security of the accumulator. The challenger runs **AccSetup** to generate the public parameters pk_{acc}. It is given pk_{acc}. Simulate the traceability game with the adversary $\mathcal{A}$, where the given pk_{acc} is used. Each query is conducted as usual using all the keys, and $\mathcal{A}$ finally outputs the forged group signature σ^*. In this case, the AHO signature on $acc^*_{V_i}$ was issued to a corrupted user. This means that $acc^*_{V_i}$ is correctly computed from the alias tokens of the user, which is denoted as V_i^*. On the other hand, the winning condition (1) states that **Verify** on the forged group signature σ^* outputs valid, which means that the verification of the accumulator for pk_{acc}, $acc^*_{V_i}$, $v^*_{i,k}$, and W^* holds. However, in case of Type 2 forgery, as above-mentioned, this signer is revoked, which means that $s_{i,k}$ corresponding to every $v_{i,k}$ in V_i^* has to be in RC due to the process of **Revoke**. On the other hand, since **RevCheck** outputs valid (because **Verify** outputs valid), $s^*_{i,k}$ corresponding to the alias token $v^*_{i,k}$ in the forged group signature σ^* is not in RC. Therefore, the alias token $v^*_{i,k}$ is not in set $V_i{}^*$. Finally, we can conclude that $v^*_{i,k} \notin V_i{}^*$ but **AccVerify** accepts pk_{acc}, $acc^*_{V_i}$, $v^*_{i,k}$ and W^* for the correctly computed $acc^*_{V_i}$ of $V_i{}^*$, which means the forgery against the accumulator. □

Table 1 Comparison of computationally expensive operations.

Scheme		Exp. in $\mathcal{G}$	Exp. in $\mathcal{T}$	Bilinear Map
Ref. [12]	**Sign**	$2m + 6$	4	3
	SignCheck	2	5	4
	RevCheck	0	0	0
Ours	**Sign**	13 (+m Mul.)	8	1
	SignCheck	0	11	11
	RevCheck	0	0	0

Table 2 Comparison of number of communicated elements.

Scheme		Elem. in Z_p	Elem. in $\mathcal{G}$	Int.
Ref. [12]	GM-Member	1	1	0
	Signer-Verifier	5	4	0
	GM-Verifier	0	0	l
Ours	GM-Member	0	8	m
	Signer-Verifier	6	9	1
	GM-Verifier	0	0	l

5. Efficiency Comparison

In this section, we compare the efficiency of our proposed scheme with the original GSPR [12].

For performance, we focus on **Sign**, **SignCheck** and **RevCheck**, since the algorithms need to be performed by the signer and the verifier, who have limited computational capabilities compared to GM. **Table 1** provides the number of operations needed in each of the algorithms for both schemes, by considering only the computationally expensive operations, i.e., exponentiation (Exp.) in $\mathcal{G}$ and $\mathcal{T}$ and bilinear map e. The original GSPR paper [12] claims that $2m$ exponentiations in $\mathcal{G}$ in Step 1 of **Sign** can be pre-computed. But, since the computation depends on the individual alias token used in each signing, we consider that the cost should be included in the signing cost. On the other hand, the bilinear map on public parameters can be pre-computed, since they are independent of each signing. Note that, in **Sign** of our scheme, m multiplications (Mul.) on $\mathcal{G}$ for the witness computation are needed in addition to the expensive operations. From this table, although the constant number of operations are increased in **Sign** and **SignCheck**, we can confirm that the signing cost including m Exp. is greatly reduced to that including only m Mul. Additionally, the good efficiency with no expensive operations in **RevCheck** is maintained.

Table 2 shows the comparison of number of elements (Elem.) in Z_p, $\mathcal{G}$, and integers (Int.) communicated in the entities. The data sizes in our proposed scheme become larger than the original GSPR. Especially, m alias tokens (integers) between GM and the

member in **Join** have to be communicated, which is not needed in the original. This is an overhead, but the size of each integer is relatively small ($\log(N \cdot m)$) and it is communicated once when joining the group, where N is the number of members.

Another trade-off for the better signing performance is the size of public key. In the original GSPR, the size is $O(m)$, but in the proposed scheme, it is $O(N \cdot m)$. Note that the public key only has to be communicated to each entity once.

6. Conclusions

In this paper, we have proposed an efficient group signature scheme with probabilistic revocation, where the signing cost is reduced from $O(m)$ exponentiations to $O(m)$ multiplications. The revocation mechanism is exactly the same as the original GSPR scheme [12], and thus the cost of revocation check is still very low. Our future works include the evaluation based on the implementation and the applications in the mobile environment.

References

[1] Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K. and Ohkubo, M.: Structure-preserving signatures and commitments to group elements, *CRYPTO 2010*, LNCS 6223, pp.209–236 (2010).

[2] Abe, M., Haralambiev, K. and Ohkubo, M.: Signing on elements in bilinear groups for modular protocol design, Cryptology ePrint Archive, Report 2010/133 (2010), available from ⟨http://eprint.iacr.org/⟩.

[3] Begum, N. and Nakanishi, T.: Efficiency Improvement in Group Signature Scheme with Probabilistic Revocation, *Proc. International Symposium on Information Theory and Its Applications 2018* (*ISITA2018*), pp.80–84 (2018).

[4] Boneh, D., Boyen, X. and Shacham, H.: Short group signatures, *CRYPTO 2004*, LNCS 3152, pp.41–55 (2004).

[5] Boneh, D. and Shacham, H.: Group signatures with verifier-local revocation, *Proc. 11th ACM Conference on Computer and Communications Security* (*ACM-CCS '04*), pp.168–177 (2004).

[6] Camenisch, J., Kohlweiss, M. and Soriente, C.: An accumulator based on bilinear maps and efficient revocation for anonymous credentials, *PKC 2009*, LNCS 5443, pp.481–500 (2009).

[7] Camenisch, J. and Lysyanskaya, A.: Dynamic accumulators and application to efficient revocation of anonymous credentials, *CRYPTO 2002*, LNCS 2442, pp.61–76 (2002).

[8] Chaum, D. and van Heijst, E.; Group signatures, *EUROCRYPT 91*, LNCS 547, pp.241–246 (1991).

[9] Emura, K. and Hayashi, T.: Road-to-vehicle communications with time-dependent anonymity: A lightweight construction and its experimental results, *IEEE Trans. Vehicular Technology*, Vol.67, No.2, pp.1582–1597 (2018).

[10] Fiat, A. and Shamir, A.: How to prove yourself: Practical solutions identification and signature problems, *CRYPTO '86*, LNCS 263, pp.186–194 (1987).

[11] Kiltz, E., Pan, J. and Wee, H.: Structure-Preserving Signatures from Standard Assumptions, Revisited, *CRYPT 2015*, Part II, LNCS 9216, pp.275–295 (2015).

[12] Kumar, V., Li, H., Park, J.J.M., Bian, K. and Yang, Y.: Group signatures with probabilistic revocation: A computationally-scalable approach for providing privacy-preserving authentication, *22nd ACM Conference on Computer and Communications Security* (*CCS 2015*), pp.1334–1345 (2015).

[13] Libert, B., Peters, T. and Yung, M.: Scalable group signatures with revocation, *EUROCRYPT 2012*, LNCS 7323, pp.609–627 (2012).

[14] Libert, B., Peters, T. and Yung, M.: Group signatures with almost-for-free revocation, *CRYPTO 2012*, LNCS 7417, pp.571–589 (2012).

[15] Libert, B., Peters, T. and Yung, M.: Short Group Signatures via Structure-Preserving Signatures: Standard Model Security from Simple Assumptions, *CRYPTO 2015*, Part II, LNCS 9216, pp.296–316 (2015).

[16] Nakanishi T. and Funabiki, N.: Verifier-local revocation group signature schemes with backward unlinkability from bilinear maps, *ASIACRYPT 2005*, LNCS 3788, pp.533–548 (2005).

[17] Nakanishi, T., Fujii, H., Hira, Y. and Funabiki, N.: Revocable group signature schemes with constant costs for signing and verifying, *PKC 2009*, LNCS 5443, pp.463–480 (2009).

[18] Neven, G., Baldini, G., Camenisch, J. and Neisse, R.: Privacy-preserving attribute-based credentials in cooperative intelligent transport systems, *2017 IEEE Vehicular Networking Conference* (*VNC 2017*), pp.131–138 (2017).

[19] Pointcheval, D. and Stern, J.: Security arguments for digital signatures and blind signatures, *J. Cryptology*, Vol.13, No.3, pp.361–396 (2000).

[20] Slamanig, D., Spreitzer, R. and Unterluggauer, T.: Linking-based revocation for group signatures: A pragmatic approach for efficient revocation checks, *Proc. Mycrypt 2016*, pp.364–388 (2016).

Nasima Begum received her Ph.D. degree in Cryptography and Information Security from Okayama University, Japan in 2014. She received her B.Sc. and M.Sc. degrees in Computer Science and Engineering from Jahangirnagar University, Dhaka, Bangladesh, in 2006 and 2010 respectively. She joined as a Lecturer at the Department of Computer Science and Engineering, Manarat International University, Dhaka, Bangladesh in January 2007. She joined as Assistant Professor at the Department of Computer Science and Engineering, University of Asia Pacific, Dhaka, Bangladesh in November 2016. She has worked as a Foreign Research Fellow in Okayama University, Japan, from October 2014 to March 2016. Her research interests include cryptography and information security, signal and image processing, and artificial intelligence. She is a member of IEEE, ACM and IEICE.

Toru Nakanishi received his M.S. and Ph.D. degrees in information and computer sciences from Osaka University, Japan, in 1995 and 2000 respectively. He joined the Department of Information Technology at Okayama University, Japan, as a research associate in 1998, and moved to the Department of Communication Network Engineering in 2000, where he became an assistant professor and an associate professor in 2003 and 2006 respectively. In 2014, he moved to the Department of Information Engineering at Hiroshima University as a professor. His research interests include cryptography and information security. He is a member of IEICE.