A Practical Group Signatures for Providing Privacy-Preserving Authentication with Revocation

Xiaohan Yue[1], Jian Xu[2], Bing Chen[1], and Yuan He[1(✉)]

[1] School of Information Science and Engineering, Shenyang University of Technology, Liaoning, People's Republic of China
isaac.y.he@ieee.org

[2] Software College, Northeastern University, Liaoning, People's Republic of China

Abstract. In recent years, many revocable group signatures schemes were proposed; however, the backward security, which can disable a revoked signer to generate group signatures pertaining to future time periods, was not fully realized through those schemes. In this paper, we present a security model with the definition of backward security and propose a revocable group signatures scheme that is more efficient than previous ones, especially in Sign and Verify algorithms, which are performed much more frequently than others. In addition, considering the heavy workload of group manager in original group signatures, we separate a group into groups by employing a decentralized model to make our scheme more scalable, and thus more practical in real-life applications.

Keywords: Group signature · Revocation · Backward security · Efficiency

1 Introduction

As a widely recognized extension to digital signatures, Chaum and Heyst proposed group signatures for the first time in 1991 [15]. With group signatures, members of a group are able to sign messages on behalf of the group while maintaining anonymity [4]. The verifier can only verify if the signature was generated by a member but cannot specify the identity of the signer. When necessary, the group manager is able to look into the signature to track the identity of the signer

We would like to sincerely thank the reviewers for their valuable comments. X. Yue was supported in part by the Program for Excellent Talents from the Department of Education of Liaoning Province under Grant LJQ2015081 and the Doctoral Research Startup Fund from the Natural Science Foundation of Liaoning Province under Grant 201601166. J. Xu was supported in part by the National Natural Science Foundation of China under Grant 61872069. Y. He was supported in part by the Natural Science Foundation of Liaoning Province under Grant 20180550194.

J. Li et al. (Eds.): SPNCE 2019, LNICST 284, pp. 226–245, 2019.
https://doi.org/10.1007/978-3-030-21373-2_18

(traceability [4]). At the same time, group signatures have non-frameability [5], that is, even the group manager cannot forge signatures of group members. These important features of group signatures have attracted many real-life applications like network identity escrow [26], online anonymous electronic voting election [35], anonymous certificate systems [14], trusted computing [12], in addition to wireless MESH networks [22] and VANET networks [44].

After group signatures were proposed [6–8,10,23], providing revocation is regarded as a major research topic, that is, an authority can revoke the membership of a user. This is called revocable group signatures and it is a very important feature for real-life applications as in many cases, a system must clearly identify the validity of a member in a timely manner to avoid any potential threat.

With revocable group signatures, there are several obvious ways of revoking a member's signature. For example, when revoking a member, the group manager can publish a new public key and provide a new signing key to each valid member, except those who have been revoked. This approach is not suitable in practice because it requires the generation of a new key and updating all members and verifiers for every such revocation. An alternative way is to revoke the member and distribute a message to existing signers. As a consequence, the signer must then prove its validity when signing. Unfortunately, this is still not considered a suitable method for revocation in real-life applications, as existing members must track the revocation message. In a word, the difficulty of providing revocation to group signatures is for the verifier to publicly confirm the status of revocation for an anonymous signer. Furthermore, the cost of such revocation is relative to the number of revoked signers so that both its communication and computation overhead can be a burden for the group manager and a performance bottleneck of the system.

To overcome such difficulties, there are some more in-depth attempts and they can be classified as follows.

- Signers are explicitly checked by the verifier for their revocation status [9,11,13,14,16,19,24,28,32,36,40,41,45].
- Revoked signers are not allowed to generate a signature to pass the verification; in this case, an explicit revocation check is not necessary [1,2,20,25,29–31,34,37–39,42].

And in the following subsection, we go through these two types of attempts in more details.

1.1 Related Works

In earlier group signature schemes with revocation, both the signing cost and the verification cost depend on the number of revoked members. In 2002, Camenisch *et al.* [14] proposed a method based on dynamic accumulator. It maps a set of values to a fixed-length string and allows for a valid membership certificate. However, this approach requires existing members to track revoked users, therefore increasing existing member's workload.

Brickell *et al.* propose a simple revocation mechanism in 2003 [11]. It is called Verifier-Local Revocation (VLR). Its formal definitions are given by Boneh and Shacham [9] and some extensions to it were proposed in [32,36,45]. In these schemes, the group member maintains a revocation list. The information of the revoked members is only sent to the verifier for verification, the signing cost is not relevant to the number of revoked group members, but the revocation list is updated every time when the group member is revoked. Therefore, the verification overhead increases as the number of revoked members increases. Recently, some VLR-type schemes have achieved sub-linear/constant verification costs [28,40,41]. However, they haven't considered backward unlinkability, that is, there are linkable parts in signatures to efficiently carry out verifications.

In 2012, a scalable scheme of the second type was presented by Libert, Peters and Yung (LPY) [31]. In this scheme, a ciphertext of broadcast encryption is periodically published while non-revoked members can decrypt this ciphertext and prove that they haven't been revoked through such decryption. As revoked signers cannot decrypt the ciphertext, they cannot generate the signature that passes the verification at early stage. For such schemes, in addition to prove its membership of the group, the signer also needs to prove that it has not been revoked. As follow-up works of the LPY scheme [1,2,29,30,37,42], revocable group signatures with compact revocation list have also been proposed.

In recent years, Ohara *et al.* [39] proposed an efficient revocable group signatures scheme to retain a constant revocation check complexity by employing the Complete Subtree (CS) method in LPY construction [31]. In this scheme, each group member is assigned to the leaf node of a tree. Instead of the identity-based encryption used in LPY, Ohara *et al.* uses the BBS signature scheme [21] in CS method, that is, signatures of nodes are written to a revocation list (RL_t), where signatures of revoked members are not in the list at a revocation epoch t. Thus, a non-revoked member can prove that its signature is in the list. In 2017, based on the method presented by Ohara *et al.* [39], Emura *et al.* [20] proposed a new revocable group signatures scheme with time-bound keys, where the notion of time-bound keys (TBK) is introduced by Chu *et al.* [16], that is, each signing key is given an expiry time. In 2018, Emura *et al.* [25] proposed and implemented a revocable group signatures scheme with scalability based on simple assumptions.

However, for security models of schemes that do not allow revoked signers to generate signatures to pass the verification, backward security is not sufficiently considered. We believe it is a security feature which is required to be explicitly defined, especially for revokable group signatures schemes relied on revocation periods. In our definition of backward security, capabilities and winning conditions of adversaries are different from other security features. When an adversary obtained private keys and credentials of all group members at a time t, he/she is able to generate a valid group signature for a revoked member who was revoked within $t^* \leq t-1$. This security feature thus ensures the rationality of a fact that when verifiers verify group signatures, it is not necessary for them to download RL_t.

In terms of performance, considering the fact that both Sign and Verify are the most frequently used algorithms for group signatures, although the above schemes [16,19,40] perform well under the random oracle model, to further improve their performance is still a good motivation for the practical application of group signatures. So this is also one of the targets of our proposal.

In addition, for previous revokable group signatures schemes [15], the group manager is responsible for issuing certificates to group members and periodically updating the revocation list for signers to download. This is inevitably a bottleneck for the deployment of group signatures at scale. Therefore, it is another concern of this work as to improve the system model of previous revokable group signatures schemes.

1.2 Our Contributions

Targeting at defects of existing work, we present a new revocable group signatures scheme in this paper. Contributions of our work are summarized as follows.

- Our scheme realizes the backward security, which disables a revoked signer to generate group signatures pertaining to future time periods. This helps complete the security model for group signatures.
- Our proposal allows deployments in a much larger scale as with its decentralized design, the group manager are freed from maintaining the revocation list while the trusted third party can be freed from the generation of group member certificates.
- Both Sign and Verify operations are highly optimized with our proposal. As both operations are dominating, we have therefore cut the computation overhead significantly.
- Security features, such as backward security, non-frameability, traceability and anonymity are fully guaranteed as our scheme is constructed with the XDH, DL and q-SDH assumptions.

Other sections are organized as below. In Sect. 2, we introduce basic knowledge of cryptography for our work. In Sect. 3, we give definitions of our scheme, its security model and our purposes. We then propose a new group signatures scheme in Sect. 4. Section 5 conducts security analyses and certifications. Comparisons between the proposed scheme and other existing schemes are made in Sect. 6. Finally, Sect. 7 concludes our work.

2 Preliminaries

In this section, we review bilinear groups, the complexity assumptions which our scheme relies on, complete sub-tree methods, and BBS+ signatures.

2.1 Bilinear Groups and Complexity Assumptions

Let G_1 and G_2 be cyclic groups of prime order p, and g_1, g_2 be generators of G_1 and G_2, respectively. Let G_T be a multiplicative cyclic group with the same order, and define $par_{Bilinear} = (p, G_1, G_2, G_T, e, g_1, g_2)$ as the set of pairing group parameters. Bilinear pair $e(G_1, G_2) \rightarrow G_T$ is a map and it satisfies properties below:

- Bilinearity: $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$ for all $a, b \in Z_p$, any $g_1 \in G_1$ and $g_2 \in G_2$.
- Non-degeneracy: $e(g_1, g_2) \neq 1$.
- Computability: The function e is efficiently computable.

Definition 1 (The Discrete Logarithm assumption (DL)). The DL assumption holds in G_1 if the probability below is negligible in security parameter κ for all adversaries $\mathcal{A}$ and all parameters $par_{Bilinear}$:

$$Adv_{\mathcal{A}}^{\mathrm{DL}}(\kappa) = Pr[x \leftarrow Z_p; u = v^x, v \leftarrow G_1 : \mathcal{A}(u, v, par_{Bilinear}) \rightarrow x]$$

Definition 2 (The Decisional Diffie-Hellman assumption). The DDH assumption holds if the probability below is negligible in security parameter κ for all adversaries $\mathcal{A}$:

$$Adv_{\mathcal{A}}^{\mathrm{DDH}}(\kappa) = Pr[\mathcal{A}(u, u^\alpha, u^\beta, z) = 1 | z = u^{\alpha \cdot \beta}] - Pr[\mathcal{A}(u, u^\alpha, u^\beta, z) = 1 | z = u^\gamma]$$

Definition 3 (The eXternal Diffie-Hellman assumption). Let $e : G_1 \times G_2 \rightarrow G_T$ be an asymmetric bilinear map, if the DDH assumption is hard in group G_1, then the XDH assumption will hold.

Definition 4 (The q-Strong Diffie-Hellman assumption). The q-SDH assumption holds if the probability below is negligible in security parameter κ, for all adversaries $\mathcal{A}$ and all parameter sets $par_{Bilinear}$:

$$\begin{aligned} Adv_{\mathcal{A}}^{q-\mathrm{SDH}}(\kappa) = Pr[x \leftarrow Z_p; g_1^x, g_1^{x^2}, ..., g_1^{x^q} \leftarrow G_1; g_2^x \\ \leftarrow G_2 : (g_1^x, g_1^{x^2}, ..., g_1^{x^q}, g_2^x, par_{Bilinear}) \rightarrow (g_1^{1/(x+c)}, c \in Z_p)] \end{aligned}$$

2.2 BBS+ Signature

The BBS+ signature scheme [21] is introduced as follows:

Given $par_{Bilinear} = (p, G_1, G_2, G_T, e)$, let $g_0, g_1, ..., g_L, g_{L+1}$ be the generators of G_1 and h be a generator of G_2.

Key Generation: Select $\gamma \leftarrow Z_p$ randomly and let $w = h^\gamma$. The secret key is $sk = \gamma$ and the verification key is $vk = w$.
Signing: For message $(m_1, ..., m_L)$, choose $\eta, \zeta \leftarrow Z_p$ randomly and compute $s = (g_0 g_1^\zeta g_2^{m_1} ... g_{L+1}^{m_L})^\lambda$ where $\lambda = (\eta + \gamma)^{-1}$. Let the signature be $\sigma = (s, \eta, \zeta)$.
Verifying: For signature $\sigma = (s, \eta, \zeta)$ and message $(m_1, ..., m_L)$, if $e(s, h^\eta vk) = e(g_0 g_1^\zeta g_2^{m_1} ... g_{L+1}^{m_L}, h)$, then the output is 1, otherwise it is 0.

This BBS+ signature scheme has unforgeability against chosen message attack (CMA) under the q-SDH assumption [8].

2.3 Complete Sub-tree Methods

In this section, we briefly introduce the Complete Subtree method [18]. Let N be the set of all signers and $R \subset N$ be the set of revoked signers. With the CS method, $N \setminus R$ is divided into m disjoint sets, that is, $N \setminus R = S_1 \cup \ ... \ \cup S_m$, where $m = O(R \cdot log(N \setminus R))$.

Definition 5 (Complete Subtree Algorithm). When a signer with index i is revoked at time t, the Complete Subtree algorithm takes the binary tree BT and a set of revoked signers R_t as inputs, where $i \in R_t$, and outputs a set of nodes. The description of CS is stated below.

$\mathrm{CS}(\mathrm{BT}, R_t)$
$\mathrm{X}, \mathrm{Y} \leftarrow \emptyset$
Add $\mathrm{Path}(i)$ to X
$\forall_x \in \mathrm{X}$
If $x_{\mathrm{left}} \notin \mathrm{X}$, then add x_{left} to Y;
If $x_{\mathrm{right}} \notin \mathrm{X}$, then add x_{right} to Y;
If $\mid RL_t \mid \ = 0$, then add root to Y;
Return Y;

In our scheme, a private key is assigned to each node of the binary tree and each user is assigned to a leaf node of the binary tree. Let $\{n_0, n_1, \ldots, n_l\}$ be the path from the root node to the leaf node where l is the height of the complete binary tree. The user then gets a key associated with each $n_i \in \{n_0, n_1, \ldots, n_l\}$ and a ciphertext is computed by keys of nodes. Let $\Theta = \{n'_0, n', \ldots, n'_m\}$ be a set of nodes and their corresponding keys are used for encryption. If the path of a user is $\{n_0, n_1, \ldots, n_l\}$, which is indicated as the authorized receiver, then there is a node ε such that $\varepsilon \in \{n_0, n_1, \ldots, n_l\} \cap \{n'_0, n', \ldots, n'_m\}$. Therefore, the user can decrypt the ciphertext using this private key corresponding to node ε.

In the proposed scheme, the Revoke and Update algorithms are constructed with the complete subtree method to ensure that non-revoked group members can generate valid signatures.

3 Definition of the Group Signatures Scheme and the Security Model

The model of our scheme is presented in Fig. 1. It consists of four entities, *i.e.*, a trusted authority (TA), the group manager (GM), group members and verifiers. Their properties are as follows:

Trusted Authority: In our scheme, The TA is responsible for maintaining system global security parameters and is trusted. If a dispute needs to be resolved, the TA has the ability to trace real identities of group members. After revealing

the actual identity, the TA can revoke malicious users and renew the revocation list (RL) which contains a set of revoked identities and a set of non-revoked tokens. We assume the revocable group signatures scheme that has its lifetime divided into epochs for revocation while at the beginning of which the trusted authority updates its revocation list.

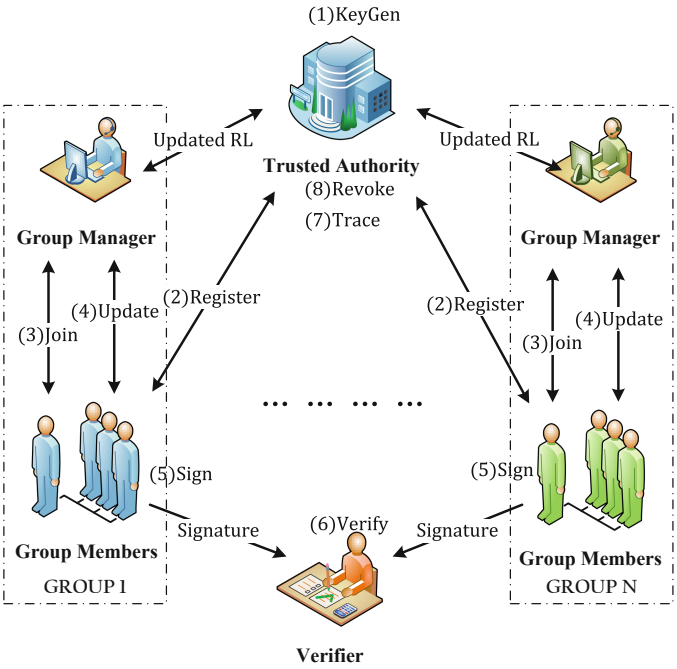

Fig. 1. The model of our scheme.

Group Manager: In our scheme, when a user wants to join a group, the GM is given the ability to generate and issue a group certificate to the user. For each group member, the GM is responsible for helping them to update the newest non-revoked tokens at a new epoch.
Group Member: Before being a group member, the user should register with TA to obtain a tag which helps him to get the non-revoked token. After joining a group, the group member is able to anonymously sign messages on behalf of the group with his secret signing key, group certificate and non-revoked token, where the non-revoked token can be updated periodically from GM.
Verifier: The verifier can verify signatures using the global public parameters and the group public key from TA or GM. When necessary, it can also forward the group signature to TA to trace a group member.

Next, we give definitions of our scheme which has seven probabilistic polynomial-time algorithms as shown in Fig. 1. Then we show security properties of our scheme.

3.1 Definitions of Our Scheme

In our proposal, the revocable group signatures scheme is made up of the following eight algorithms/protocols.

(1) KeyGen algorithm
KeyGen(λ): Every entity in our scheme performs this algorithm.

- **TAKg**(λ): TA executes this algorithm at the setup stage with a security parameter $\lambda \in \mathbb{N}$ as the input to produce the global public key gpk, the secret tracing key tk_{TA} and the non-revoked token secret key rk_{TA}. After such key generations, TA then publishes gpk.
- **GMKg**(gpk): Taking gpk as the input, it generates the secret key sk_{GM} and the group public key pk_{GM} of the group manager.
- **UKg**(gpk): Every new user before interacting with TA and GM executes this algorithm. Taking gpk as the input, it generates the secret signing key sk_M and the personal public key pk_M for a user.

(2) Register protocol
Register(User:(sk_M, pk_M),TA:($BinaryTree$)): Before joining a group, each user needs to interact with TA to prove the knowledge of private key sk_M with some zero-knowledge proof protocol. TA will then assign a tag_i to user i.

(3) Join protocol
Join(User:(gpk, sk_M, pk_M),GM:($gpk, pk_{GM}, sk_{GM}, \text{RL}_t$)): It is interactive between the group manager and a user when the latter turns into a member. For the first time when a user joins a group, this protocol terminates with the user i obtaining a group membership certificate gCert_i. It is to be noted that any revoked member in the revoked user set R_t is not allowed to join a group.

(4) Update algorithm
For each revocation epoch, group members need to update their non-revoked tokens by communicating with the GM giving an assumption that the GM can obtain the current revocation list RL_t.
Update(Member:(tag),GM:(RL_t)): At a revocation epoch t, a member i sends a request to the GM to obtain a new non-revoked token. When receiving such request, the GM checks if member i is in R_t. If not, the GM forms a token $\text{token}_{i,t}$ from the set Φ in RL_t according to the member's tag_i, and then returns it to the member.

(5) Sign algorithm
Sign($gpk, t, token_{i,t}, \text{gCert}_i, sk_M, msg$): When being given an epoch t with an updated $\text{token}_{i,t}$, a group membership certificate gCert_i, a secret signing key sk_M, and a message msg, this algorithm will generate a group signature σ.

(6) Verify algorithm
Verify($\sigma, msg, t, gpk, pk_{GM}$): When being given a signature σ, a revocation epoch t, a message msg, the global public key gpk, and the group public key pk_{GM}, this deterministic algorithm will output either 0 (if invalid) or 1 (if valid).

(7) Trace algorithm
Trace$(gpk, tk_{TA}, \mathrm{RL}_t, msg, \sigma)$: This deterministic verifiable algorithm traces a signer by taking gpk, the tracing key tk_{TA}, RL_t, and the message-signature pair (msg, σ) as inputs while returning the identity of this signer i.

(8) Revoke algorithm
Revoke$(gpk, rk_{TA}, t, \mathrm{R}_t)$: This algorithm allows TA to generate an updated revocation list $\mathrm{RL}_t = \{t, \mathrm{R}_t, \Phi = \{\mathrm{token}_{i,t}\}_{i=0}^{m}\}$ for a new revocation epoch t, where the token set Φ has all tokens of non-revoked users. It accepts the set R_t of identities of revoked members for gpk, rk_{TA} and revocation epoch t as inputs while it outputs the revocation list RL_t for epoch t.

3.2 The Security Model

Here, we introduce security properties of our scheme. First, notations and the oracles used in the definitions are given as follows:

$\mathcal{O}_{\mathrm{a-join}}$: The adversary $\mathcal{A}$ executes Join algorithm with honest group manager, and the member that collude with the adversary is added to the group. Then the number of members is incremented and adds the information of new member to a registration table Reg.

$\mathcal{O}_{\mathrm{b-join}}$: The adversary $\mathcal{A}$ executes Join algorithm while colluding the group manager (this member does not collude with the adversary). Then the number of members is incremented and adds the information of new member to a registration table Reg.

$\mathcal{O}_{\mathrm{AddM}}$: By calling this oracle with an argument, an identity i, the adversary can add i to the group as an honest user where HM is the set of honest members. It also picks a pair of personal public and private keys $(sk_{M,i}, pk_{M,i})$ for i. It then executes the Join protocol (on behalf of i). When finished, it adds the information of i to the registration table Reg. The calling adversary then receives $pk_{M,i}$.

$\mathcal{O}_{\mathrm{Sign}}$: It receives a query that is a message msg and identity i and returns $\perp$ if $i \notin$ HM, and otherwise returns σ for the member i and epoch t.

$\mathcal{O}_{\mathrm{Trace}}$: The adversary $\mathcal{A}$ can call this oracle with arguments, a message msg and signature σ, to obtain the output of the Trace algorithm on msg, σ, computed under the tracing key tk_{TA} given that σ was not previously returned as a response for any query to $\mathcal{O}_{\mathrm{Ch}}$.

$\mathcal{O}_{\mathrm{Ch}}$: On input $i_0, i_1 \in$ HM, and a message msg, the challenge oracle computes signature σ by performing the Sign algorithm with the private signing key of i_b, where $b \in_R \{0, 1\}$, and returns σ. The oracle keeps as record the message-signature pair to make sure that the adversary does not call the tracing oracle on it later.

$\mathcal{O}_{\mathrm{MSK}}$: On input i, the member secret key oracle reveals $(sk_{M,i}, pk_{M,i})$ and adds i to the set CM of corrupted members.

$\mathcal{O}_{\mathrm{WReg}}$: On input i, the oracle allows the adversary to modify member i in the registration table Reg.

$\mathcal{O}_{\mathrm{RReg}}$: On input i, the oracle allows the adversary to read member i in the registration table Reg.

$\mathcal{O}_{\mathrm{RRL}}$: On input i and t^*, the oracle allows the adversary to read member i of the revocation List RL_{t^*} at an epoch t^*.

$\mathcal{O}_{\mathrm{Update}}$: At an epoch t, the adversary can call this oracle with an argument, an identity i, to obtain the corresponding non-revoked token $\mathrm{token}_{i,t}$.

$\mathcal{O}_{\mathrm{Revoke}}$: It revokes a member from the group. It receives a query of member identity $i \in \mathrm{HM}$ and increments t, adds i to RL_t and updates RL_t.

Next, we define the anonymity with backward unlinkability (BU-anonymity), which guarantees that no adversary (who does not have tk_{TA}) can distinguish if signers (even if the signer has been revoked) of two group signatures are the same.

Definition 1: BU-Anonymity is defined by the following game.
Attack-Game $\mathrm{Game}_{\mathcal{A}}^{bu-anon}(\lambda)$:

$b \in_R \{0,1\}$
$(gpk, pk_{GM}, sk_{GM}, rk_{TA}, tk_{TA}) \leftarrow \mathrm{KeyGen}(\lambda)$
$\mathrm{CM} \leftarrow \emptyset, \mathrm{HM} \leftarrow \emptyset, \mathrm{RL}_t \leftarrow \emptyset$
$d \leftarrow \mathcal{A}(gpk, pk_{GM}, sk_{GM} : \mathcal{O}_{\mathrm{Ch}}, \mathcal{O}_{\mathrm{b-join}}, \mathcal{O}_{\mathrm{WReg}}, \mathcal{O}_{\mathrm{MSK}}, \mathcal{O}_{\mathrm{Trace}}, \mathcal{O}_{\mathrm{Revoke}}, \mathcal{O}_{\mathrm{Update}}, \mathcal{O}_{\mathrm{Sign}})$
If $d = b$ return 1 else 0.

The advantage of the adversary $\mathcal{A}$ against this game is $\mathrm{Adv}_{\mathcal{A}}^{bu-anon}(\lambda) = \mid Pr[\mathrm{Game}_{\mathcal{A}}^{bu-anon}(\lambda)] - 1/2 \mid$. We say that our scheme satisfies BU-anonymity if $\mathrm{Adv}_{\mathcal{A}}^{bu-anon}(\lambda)$ is negligible in λ for any probabilistic polynomial-time algorithm $\mathcal{A}$.

Next, we define non-frameability which guarantees that no adversary (who can corrupt the GM and the TA) is able to produce group signatures with its tracing result being an honest user.

Definition 2: Non-frameability is defined by the following game.
Attack-Game $\mathrm{Game}_{\mathcal{A}}^{frame}(\lambda)$:

$(gpk, pk_{GM}, sk_{GM}, rk_{TA}, tk_{TA}) \leftarrow \mathrm{KeyGen}(\lambda)$
$\mathrm{CM} \leftarrow \emptyset, \mathrm{HM} \leftarrow \emptyset, \mathrm{RL}_t \leftarrow \emptyset$
$(msg, \sigma, t, \mathrm{RL}_t) \leftarrow \mathcal{A}(gpk, pk_{GM}, sk_{GM}, tk_{TA} : \mathcal{O}_{\mathrm{b-join}}, \mathcal{O}_{\mathrm{WReg}}, \mathcal{O}_{\mathrm{Sign}}, \mathcal{O}_{\mathrm{Revoke}}, \mathcal{O}_{\mathrm{Update}})$
If $\mathrm{Verify}(gpk, pk_{GM}, msg, \sigma, t) = 0$, return 0.
$i \leftarrow \mathrm{Trace}(gpk, rk_{TA}, t, \mathrm{RL}_t, msg, \sigma)$
If $i \in \mathrm{HM}$, return 1.
Reurn 0.

The advantage of the adversary $\mathcal{A}$ against this game is $\mathrm{Adv}_{\mathcal{A}}^{frame}(\lambda) = Pr[\mathrm{Game}_{\mathcal{A}}^{frame}(\lambda) = 1]$. We say that our scheme satisfies non-frameability if $\mathrm{Adv}_{\mathcal{A}}^{frame}(\lambda)$ is negligible in λ for any probabilistic polynomial-time algorithm $\mathcal{A}$.

Next, we define traceability which guarantees that no adversary (who does not have sk_{GM}) can generate a valid group signature with its tracing result being outside of the set of non-revoked adversarially-controlled users.

Definition 3: Traceability is defined by the following game.

Attack-Game $\mathrm{Game}_{\mathcal{A}}^{trace}(\lambda)$:

$(gpk, pk_{GM}, sk_{GM}, rk_{TA}, tk_{TA}) \leftarrow \mathrm{KeyGen}(\lambda)$
$\mathrm{CM} \leftarrow \emptyset, \mathrm{HM} \leftarrow \emptyset, \mathrm{RL}_t \leftarrow \emptyset$
$(msg, \sigma, t) \leftarrow \mathcal{A}(gpk, pk_{GM}, tk_{TA} : \mathcal{O}_{\mathrm{a-join}}, \mathcal{O}_{\mathrm{AddM}}, \mathcal{O}_{\mathrm{MSK}}, \mathcal{O}_{\mathrm{RReg}}, \mathcal{O}_{\mathrm{Sign}}, \mathcal{O}_{\mathrm{Revoke}}, \mathcal{O}_{\mathrm{Update}})$
If $\mathrm{Verify}(gpk, pk_{GM}, t, msg, \sigma) = 0$, return 0.
$i \leftarrow \mathrm{Trace}(pk_{GM}, tk_{TA}, reg, msg, \sigma, \mathrm{RL}_t)$
If $i \notin \mathrm{HM} \cup \mathrm{CM} \backslash \mathrm{R}_t$, return 1.
Reurn 0.

The advantage of the adversary $\mathcal{A}$ against this game is $\mathrm{Adv}_{\mathcal{A}}^{trace}(\lambda) = Pr[\mathrm{Game}_{\mathcal{A}}^{trace}(\lambda) = 1]$. We say that our scheme satisfies traceability if $\mathrm{Adv}_{\mathcal{A}}^{frame}(\lambda)$ is negligible in λ for any probabilistic polynomial-time algorithm $\mathcal{A}$.

Next, we define backward security which guarantees that no adversary (who does not have rk_{TA}) can forge a valid group signature with its tracing result being in the set of revoked users.

Definition 4: Backward Security is defined by the following game.

Attack-Game $\mathrm{Game}_{\mathcal{A}}^{backward}(\lambda)$:

$(gpk, pk_{GM}, sk_{GM}, rk_{TA}, tk_{TA}) \leftarrow \mathrm{KeyGen}(\lambda)$
$\mathrm{CM} \leftarrow \emptyset, \mathrm{HM} \leftarrow \emptyset, \mathrm{RL}_t \leftarrow \emptyset$
$(msg, \sigma, t) \leftarrow \mathcal{A}(gpk, pk_{GM}, sk_{GM}, tk_{TA} : \mathcal{O}_{\mathrm{a-join}}, \mathcal{O}_{\mathrm{AddM}}, \mathcal{O}_{\mathrm{MSK}}, \mathcal{O}_{\mathrm{Sign}}, \mathcal{O}_{\mathrm{RRL}}, \mathcal{O}_{\mathrm{Revoke}}, \mathcal{O}_{\mathrm{Update}})$
If $\mathrm{Verify}(gpk, pk_{GM}, t, msg, \sigma) = 0$, return 0.
$i \leftarrow \mathrm{Trace}(pk_{GM}, tk_{TA}, reg, msg, \sigma, \mathrm{RL}_t)$
If $i \in \mathrm{R}_t$, return 1.
Reurn 0.

The advantage of the adversary $\mathcal{A}$ against this game is $\mathrm{Adv}_{\mathcal{A}}^{backward}(\lambda) = Pr[\mathrm{Game}_{\mathcal{A}}^{backward}(\lambda) = 1]$. We say that our scheme satisfies backward security if $\mathrm{Adv}_{\mathcal{A}}^{backward}(\lambda)$ is negligible in λ for any probabilistic polynomial-time algorithm $\mathcal{A}$.

4 The Proposed Group Signatures Scheme

In this section, we present details of our group signatures scheme. This scheme is based on an assumption that interactions between new users and the TA/GM happen through a secure channel.

KeyGen(λ): In this algorithm, TA generates a private key for itself and parameters for the system. In addition, GM and all the users also generate private key pairs for themselves. Details are stated below:

- **TAKg**(λ): TA carries out steps below with the security parameter λ:
  - Choose an asymmetric bilinear group pair $(G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle)$ of prime order $p \in \{0,1\}^\lambda$ and a pairing function $e : G_1 \times G_2 \to G_T$.
  - Select $\dot{g}_1, \ddot{g}_1 \leftarrow G_1$ randomly and a secure cryptographic hash function H where $\mathrm{H}(\cdot) : \{0,1\}^* \to Z_p$.
  - Select a secret key $\gamma \leftarrow Z_p$ randomly, and issue $(rk_{TA}, pk_{TA}) = (\gamma, g_2^\gamma)$.
  - Select $x_1', x_2', y_1', y_2' \leftarrow Z_p$ randomly, and compute $\varphi_1 = \dot{g}_1^{x_1'} \ddot{g}_1^{x_2'}, \varphi_2 = \dot{g}_1^{y_1'} \ddot{g}_1^{y_2'}$.
  - Select a secret key $\nu \leftarrow Z_p$ randomly, and compute $u = \dot{g}_1^\nu$. Let $tk_{TA} = \nu$ be the tracing key of the TA.
  - Keep rk_{TA} and tk_{TA} secret.
  - Publish global public system parameters $gpk = (p, G_1, G_2, G_T, e, g_1, g_2, \dot{g}_1, \ddot{g}_1, \varphi_1, \varphi_2, u, \mathrm{H}, pk_{TA})$.
- **GMKg**(gpk): Each GM takes steps below:
  - Select a secret key $\omega \leftarrow Z_p$ randomly, and compute $(sk_{GM}, pk_{GM}) = (\omega, g_2^\omega)$. pk_{GM} is the group public key and sk_{GM} is the private key of GM.
  - Keep sk_{GM} secret.
- **UKg**(gpk): Users take steps below:
  - Select a secret key $\chi \leftarrow Z_p$ randomly, and compute $(sk_M, pk_M) = (\chi, \ddot{g}_1^\chi)$, χ is the private key of a user.
  - Keep sk_M secret.

Register(User:(sk_M, pk_M),TA:$(BinaryTree)$): This protocol is based on an assumption that interactions between new users and the TA is carried out in a secure channel.

- Users interact with the TA with some zero-knowledge proof protocol to obtain the tag used for getting the non-revoked token later.
  - In order to prove the knowledge of the secret signing key $sk_{M,i} = \chi_i$, the user i randomly chooses $\gamma_\chi \leftarrow Z_p$, and computes $R_\chi = \ddot{g}_1^{\gamma_\chi}$, $c = \mathrm{H}(gpk \parallel pk_{M,i} \parallel R_\chi)$ and $s_\chi = \gamma_\chi + c \cdot \chi_i$.
  - Through the above operations, the user sends the proof $(pk_{M,i}, c, s_\chi)$ to TA.
  - TA computes $\check{R}_\chi = \ddot{g}_1^{s_\chi} pk_M^c$, and checks whether that $c = \mathrm{H}(gpk \parallel pk_{M,i} \parallel \check{R}_\chi)$. The proof is valid if positive and this means that user i knows the knowledge of the secret key χ_i.
- According to the CS method in Sect. 2.3, the TA will give user i an available leaf node ν_i of the binary tree and a path $\rho_i := \langle n_0 = \epsilon, n_1, \ldots, n_l = \nu_i \rangle$ connecting the leaf ν_i to the root ϵ. The TA will then send a path ρ_i to user i. Here we name ρ_i as tag tag_i of i, which is used to request the non-revoked token from the GM.

– Next, the TA will carry out the Revoke algorithm to update the set of non-revoked tokens and store $\{i, pk_{M,i}, \text{tag}_i, *\}$ in the registration table Reg, where the symbol $*$ denotes the group membership certificates of user i and it will be generated by the Join protocol.

Join(User:(gpk, sk_M, pk_M),GM:$(gpk, pk_{GM}, sk_{GM}, \text{RL}_t)$): This protocol is interactive between the GM and users.

– To obtain a group membership certificate from the GM, at first, the user need to perform a zero-knowledge proof protocol as described in Register to prove the knowledge of its secret signing key.
– Next, the user sends a node $\varepsilon \in$ tag to the GM (The node ε was determined in Update).
– After receiving the node ε, the GM takes steps below.
  • Checks whether user i exists in the R_t. If positive, move on; otherwise, abort this step.
  • Compute the group membership certificate $\text{gCert}_i = (g_1 \dot{g}_1^{\varepsilon} \cdot pk_{M,i})^{1/(sk_{GM}+\eta)}$, where $\eta \in_R Z_p$.
  • Send $(\text{gCert}_\text{i}, \eta)$ to the user.
  • Send a copy of the user's $(i, pk_{M,i}, \text{gCert}_i)$ to the TA who will update the registration information of user i with the copy.
  • After receiving the response $(\text{gCert}_\text{i}, \eta)$ from the GM, user i checks if $e(\text{gCert}_i, pk_{GM} \cdot g_2^{\eta}) = e(pk_{M,i}, g_2) \cdot e(g_1 \dot{g}_1^{\varepsilon}, g_2)$. If yes, the user accepts its group certificate $\text{gCert}_i = (g_1 \dot{g}_1^{\varepsilon} \ddot{g}_1^{\chi})^{1/(\omega+\eta)}$.

Sign$(gpk, t, token_{i,t}, \text{gCert}_i, sk_M, msg)$: Upon entering $msg \in \{0,1\}^*$, the GM signs it with SIGN and sends the signature with msg. Details are stated below:

– Select $\zeta \leftarrow Z_p$ randomly, and output: $\psi_1 = \text{gCert} \cdot u^{\zeta}$, $\psi_2 = \text{token}_{\varepsilon,t} \cdot g_1^{\zeta}$, $\psi_3 = \dot{g}_1^{\zeta}$, $\psi_4 = \ddot{g}_1^{\zeta}$, $\psi_5 = (\varphi_1 \varphi_2^h)^{\zeta}$, where $h = \text{H}(\psi_1 \parallel \psi_2 \parallel \psi_3 \parallel \psi_4)$. Let $\alpha = \zeta \cdot \eta$ and $\beta = \zeta \cdot \eta'$.
– Compute the signature of knowledge (SPK) as below. $V = SPK\{(\zeta, \alpha, \beta, \varepsilon, \chi, \eta, \eta'):$
$e(\psi_1, g_2)^{-\eta} e(\ddot{g}_1, g_2)^{\chi} e(u, g_2)^{\alpha} e(u, pk_{GM})^{\zeta} e(\dot{g}_1, g_2)^{\varepsilon} = e(\psi_1, pk_{GM})/e(g_1, g_2)$,
$e(\psi_2, g_2)^{-\eta'} e(\ddot{g}_1, g_2)^{t} e(g_1, g_2)^{\beta} e(g_1, pk_{TA})^{\zeta} e(\dot{g}_1, g_2)^{\varepsilon} = e(\psi_2, pk_{TA})/e(g_1, g_2)$,
$\psi_3 = \dot{g}_1^{\zeta}$,$\psi_4 = \ddot{g}_1^{\zeta}$,$\psi_5 = (\varphi_1 \varphi_2^h)^{\zeta}\}(msg)$
The SPK is computed using the following steps.
  • Pick blind factors $r_\alpha, r_\beta, r_\zeta, r_\varepsilon, r_\chi, r_\eta, r_{\eta'} \leftarrow Z_p$ and compute:
    $R_1 \leftarrow \dot{g}_1^{r_\zeta}$
    $R_2 \leftarrow \ddot{g}_1^{r_\zeta}$
    $R_3 \leftarrow (\varphi_1 \varphi_2^h)^{r_\zeta}$
    $R_{\text{gCert}} \leftarrow e(\dot{g}_1, g_2)^{r_\varepsilon} e(\psi_1, g_2)^{-r_\eta} e(\ddot{g}_1, g_2)^{r_\chi} \cdot e(u, g_2)^{r_\alpha} e(u, pk_{GM})^{r_\zeta}$
    $R_{\text{token}} \leftarrow e(\dot{g}_1, g_2)^{r_\varepsilon} e(\psi_2, g_2)^{-r_{\eta'}} e(\ddot{g}_1, g_2)^{t} \cdot e(g_1, g_2)^{r_\beta} e(g_1, pk_{TA})^{r_\zeta}$
  • Compute $c = \text{H}(msg \parallel \psi_1 \parallel \psi_2 \parallel \psi_3 \parallel \psi_4 \parallel \psi_5 \parallel R_1 \parallel R_2 \parallel R_3 \parallel R_{\text{gCert}} \parallel R_{\text{token}})$.

- With results above, values below are computed.
  $s_\alpha \leftarrow r_\alpha + c \cdot \alpha$
  $s_\beta \leftarrow r_\beta + c \cdot \beta$
  $s_\zeta \leftarrow r_\zeta + c \cdot \zeta$
  $s_\eta \leftarrow r_\eta + c \cdot \eta$
  $s_{\eta'} \leftarrow r_{\eta'} + c \cdot \eta'$
  $s_\chi \leftarrow r_\chi + c \cdot \chi$
  $s_\varepsilon \leftarrow r_\varepsilon + c \cdot \varepsilon$

  The group signature is $\sigma = (c, s_\alpha, s_\beta, s_\eta, s_\zeta, s_{\eta'}, s_\chi, s_\varepsilon, \psi_1, \psi_2, \psi_3, \psi_4, \psi_5)$.

Verify$(\sigma, msg, t, gpk, pk_{GM})$: The verifier executes Verify to validate the received message (msg, σ). Details are stated below:

- Compute values below.
  $\check{R}_{\text{gCert}} \leftarrow e(\dot{g}_1, g_2)^{s_\varepsilon} e(\psi_1, g_2)^{-s_\eta} e(\ddot{g}_1, g_2)^{s_\chi} e(u, g_2)^{s_\alpha} e(u, pk_{GM})^{s_\zeta} (e(g_1, g_2)/e(\psi_1, pk_{GM}))^c$
  $\check{R}_{\text{token}} \leftarrow e(\dot{g}_1, g_2)^{s_\varepsilon} e(\psi_2, g_2)^{-s_{\eta'}} e(g_1, g_2)^{s_\beta} e(g_1, pk_{TA})^{s_\zeta} (e(\ddot{g}_1^t g_1, g_2)/e(\psi_2, pk_{TA}))^c$
  $\check{R}_1 \leftarrow \dot{g}_1^{s_\zeta} \psi_3^{-c}$
  $\check{R}_2 \leftarrow \ddot{g}_1^{s_\zeta} \psi_4^{-c}$
  $\check{R}_3 \leftarrow (\varphi_1, \varphi_2^h)^{s_\zeta} \psi_5^{-c}, whereh = \mathrm{H}(\psi_1 \parallel \psi_2 \parallel \psi_3 \parallel \psi_4)$.
- Check whether $c = \mathrm{H}(msg \parallel \psi_1 \parallel \psi_2 \parallel \psi_3 \parallel \psi_4 \parallel \psi_5 \parallel \check{R}_1 \parallel \check{R}_2 \parallel \check{R}_3 \parallel \check{R}_{\text{gCert}} \parallel \check{R}_{\text{token}})$. The verifier will accept the message if the equation holds; otherwise the message will be rejected.

Trace$(gpk, tk_{TA}, \mathrm{RL}_t, msg, \sigma)$: It is possible to identify the actual signer with the valid group signature σ on message msg. The steps are as follows.

- Compute the group certificate $\text{gCert} = \psi_1 / \psi_3^{tk_{TA}}$.
- Look up gCert in the registration table and retrieve its registration information $\{i, pk_{M,i}, tag_i, \text{gCert}\}$. If matched, return i. Otherwise, output 0 and abort with a failure.

Revoke$(gpk, rk_{TA}, t, \mathrm{R}_t)$: In general, the TA periodically updates RL_t; or it updates when a member is revoked. Meanwhile, the TA will also update the non-revoked token set Φ, and the revoked user set R_t with an epoch t. Details are presented below.

- Determine the non-revoked node set $\Theta = \{n_0, n_1, \ldots, n_m\}$, where m is the number of non-revoked users, with the CS covering algorithm.
- For $i = 1$ to m, select $\eta_i' \leftarrow Z_p$ randomly, and output the non-revoked token $(\text{token}_{i,t} = (g_1 \dot{g}_1^{n_i} \ddot{g}_1^t)^{1/(rk_{TA} + \eta_i')}, \eta_i')$.
- Send the updated $\mathrm{RL}_t = \{t, \mathrm{R}_t, \{\text{token}_{i,t}\}_{i=0}^m, \Theta\}$ to each GM via an authenticated and secure channel.

Update(Member:(tag),GM:(RL_t)): Group members periodically update their non-revoked tokens from the GM. Interactions between group members and the GM are presented below.

- The member forwards pk_M and tag to GM.
- Upon receiving pk_M and tag, the GM performs steps as follows:
  - Find out if member i exists in R_t. If no, move on; otherwise, abort.
  - Select a node ε from the intersection of the tag and the node set Θ, as in Sect. 2.3.
  - Search in the non-revoked token set $\Phi = \{\text{token}_i\}_{i=1}^m$ in RL_t for $\text{token}_{\varepsilon,t}$ corresponding to node ε.
  - Encrypt the selected $\text{token}_{\varepsilon,t}$ and ε for the member.
- After the response is received, the member checks if $e((g_1\dot{g}_1^{\varepsilon}\ddot{g}_1^t)^{1/(rk_{TA}+\eta_i')}, pk_{TA} \cdot g_2^{\eta'}) = e(\ddot{g}_1^t, g_2) \cdot e(g_1\dot{g}_1^{\varepsilon}, g_2)$. If yes, the member accepts it.

5 Security Analyses

Here, we discuss the security of our scheme. That is, we explain that our scheme satisfies backward security, BU-anonymity, non-frameability, traceability defined in Sect. 3.1. For backward security, the attack on backward security is to fake a BBS+ signature as a non-revoked token. Therefore, the security against backward security attacks can be simplified as the unforgeability of the BBS+ signature scheme, which was proved in [21]. For traceability, the adversary is essentially concerned with faking a valid group membership certificate. This also can be reduced to the unforgeability of the BBS+ scheme. For security against framing attacks, in the join protocol of our proposal, a user chooses its secret signing key sk_{M} while the GM does not know it. Therefore, from a forged signature output by the adversary of framing attacks, an algorithm can be constructed so that it extracts this secret key and solves the DL problem with it. Furthermore, because of the CCA security of the Cramer-Shoup encryption scheme [17], our scheme is anonymous. Due to space constraints, we will present security proofs of the following theorems in an extended version of this paper.

Theorem 1. *The proposed group signatures scheme has BU-anonymity in the random oracle model if the XDH assumption holds in* (G_1, G_2, G_T).

Theorem 2. *The proposed group signatures scheme has non-frameability in the random oracle model under the DL assumption.*

Theorem 3. *The proposed group signatures scheme has traceability in the random oracle model under the q-SDH assumption.*

Theorem 4. *The proposed group signatures scheme has backward security in the random oracle model under the q-SDH assumption.*

6 Performance Evaluations

We evaluate the computation and communication overhead of our scheme in this section and compare its performance to two existing schemes [20,39]. Our scheme is implemented with the Barreto-Naehrig (BN) curves [3] over a 382-bit prime field to ensure 128-bit security [27,43]. Our implementation was simulated on a workstation with specifications as follows:

- CPU: Intel Xeon E5-2680v2 (3.6 GHz)
- OS: Windows Server 2012
- Compilation environment: Microsoft Visual Studio C++ 2017
- Crypto-library: the PBC library [33]

For above-mentioned experimental settings, Table 1 summarizes the benchmarks used in our work.

Table 1. Benchmarks of group operations on a 382-bit BN curve.

Operations	Time (μsec)
Mul($\mathbb{G}_1$)	344.5
Mul($\mathbb{G}_2$)	471.3
Exp($\mathbb{G}_T$)	981.6
$\mathbb{P}$	1847.3

Here, Mul($\mathbb{G}_1$), Mul($\mathbb{G}_2$), Exp($\mathbb{G}_T$) are scalar multiplication on G_1, G_2, and exponentiation on G_T, respectively. $\mathbb{P}$ is the time to perform a pairing operation. We only consider the influence of these four operations as the speed of signature generation/verification actually depends on them. We firstly consider the load of computation for our algorithms according to benchmarks of the PBC library. We reduce the computation overhead of the Sign/Verify algorithms by decreasing the number of exponentiations on G_T.

For the Sign algorithm, we transform R_{gCert} and R_{token} as $R_{\text{gCert}} \leftarrow e(\dot{g}_1^{r_\varepsilon}\ddot{g}_1^{r_\chi}u^{r_\alpha}\psi_1^{-r_\eta}, g_2)e(u^{r_\zeta}, pk_{GM})$, $R_{\text{token}} \leftarrow e(\dot{g}_1^{r_\varepsilon}\ddot{g}_1^{t}g_1^{r_\beta}\psi_2^{-r_{\eta'}}, g_2)e(g_1^{r_\zeta}, pk_{TA})$. By precomputing pairing values, such as $e(\dot{g}_1, g_2)$, original computations require 2Mul($\mathbb{G}_1$)+7Exp($\mathbb{G}_T$)+2$\mathbb{P}$. But 9Mul($\mathbb{G}_1$)+4$\mathbb{P}$ are required in our modifications.

For the Verify algorithm, we also transform $\check{R}_{\text{gCert}}$ and $\check{R}_{\text{token}}$ as $\check{R}_{\text{gCert}} \leftarrow e(\dot{g}_1^{s_\varepsilon}\psi_1^{-s_\eta}\ddot{g}_1^{s_\chi}u^{s_\alpha}g_1^{c}, g_2)e(u^{s_\zeta}\psi_1^{-c}, pk_{GM})$, $\check{R}_{\text{token}} \leftarrow e(\dot{g}_1^{s_\varepsilon}\psi_2^{-s_{\eta'}}\ddot{g}_1^{tc}g_1^{c+s_\beta}, g_2)$ $e(\psi_2^{-c}g_1^{s_\zeta}, pk_{TA})$ which originally requires 5Mul($\mathbb{G}_1$)+7Exp($\mathbb{G}_T$)+5$\mathbb{P}$ with precomputed pairing values, but our modifications require 12Mul($\mathbb{G}_1$)+4$\mathbb{P}$. In Table 2, we present comparisons of the computation overhead between our proposal and schemes in [39] and [20]. Our attention was laid on two specific algorithms: Sign (signature generation algorithm) and Verify (signature correctness verifying algorithm). We focus on them as they need to be frequently performed by the GM and the verifier.

Table 2. Comparisons of the computation overhead.

Schemes	Sign (opt.)	Verify (opt.)
Ohara *et al.* [39]	$20\text{Mul}(\mathbb{G}_1)+1\text{Exp}(\mathbb{G}_2)+4\mathbb{P}$	$21\text{Mul}(\mathbb{G}_1)+6\mathbb{P}$
Emura *et al.* [20]	$28\text{Mul}(\mathbb{G}_1)+2\text{Exp}(\mathbb{G}_T)+4\mathbb{P}$	$20\text{Mul}(\mathbb{G}_1)+12\text{Exp}(\mathbb{G}_2)+2\text{Mul}(\mathbb{G}_T)+8\mathbb{P}$
Proposed	$17\text{Mul}(\mathbb{G}_1)+4\mathbb{P}$	$18\text{Mul}(\mathbb{G}_1)+4\mathbb{P}$

In Table 2, It can be seen that our proposal takes low computational cost among the existing schemes to perform signature generation and signature verification processes. For Sign algorithm, the proposed scheme can generate a group signature with 13.25 ms whereas the other two schemes [39] and [20] take 14.75 ms and 19 ms respectively. When verifying a signature, our proposal takes 13.6 ms but the schemes [39] and [20] take 18.3 ms and 32.73 ms.

Table 3. Comparisons of the communication overhead.

Schemes	Element Size	Signature Length
Ohara *et al.* [39]	$20\mathbb{G}_1+11\mathbb{Z}_p$	871 bytes
Emura *et al.* [20]	$12\mathbb{G}_1+4\mathbb{Z}_p$	780 bytes
Proposed	$5\mathbb{G}_1+8\mathbb{Z}_p$	629 bytes

The communication overhead is presented in TABLE 3. With our proposal, a group signature has 13 group elements (5 elements in G_1 and 8 elements in Z_p). On the other hand, in the schemes [39] and [20], each signature has 18 group elements (7 elements in G_1 and 11 elements in Z_p) or 16 group elements (12 elements in G_1 and 4 elements in Z_p). When BN&382-bit is employed, the size of a value in Z_p, an element in G_1, an element in G_2 and an element in G_T are 48 bytes, 49 bytes, 97 bytes and 384 bytes, respectively. Therefore, the signature length in our proposal is 629 bytes and it is smaller than the other two schemes.

7 Conclusions

In this paper, we have further improved revocable group signatures with respect to signature generations/verifications and the signature size. In our security model, we present a new security feature, backward security; we believe this feature is necessary for revocable group signature schemes as it ensures unforgeability of group signatures when group members were revoked and rationality for verifications without the RL, especially for LPY-type schemes. For real-life applications, our scheme applies a decentralized group model to relax the original group manager from the heavy workload of revocation list maintenance which makes the deployment of group signatures more practical in providing privacy-preserving authentications.

References

1. Attrapadung, N., Emura, K., Hanaoka, G., Sakai, Y.: A revocable group signature scheme from identity-based revocation techniques: achieving constant-size revocation list. In: Boureanu, I., Owesarski, P., Vaudenay, S. (eds.) ACNS 2014. LNCS, vol. 8479, pp. 419–437. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07536-5_25
2. Attrapadung, N., Emura, K., Hanaoka, G., Sakai, Y.: Revocable group signature with constant-size revocation list. Comput. J. **58**(10), 2698–2715 (2015)
3. Barreto, P.S.L.M., Naehrig, M.: Pairing-friendly elliptic curves of prime order. In: Preneel, B., Tavares, S. (eds.) SAC 2005. LNCS, vol. 3897, pp. 319–331. Springer, Heidelberg (2006). https://doi.org/10.1007/11693383_22
4. Bellare, M., Micciancio, D., Warinschi, B.: Foundations of group signatures: formal definitions, simplified requirements, and a construction based on general assumptions. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 614–629. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-39200-9_38
5. Bellare, M., Shi, H., Zhang, C.: Foundations of group signatures: the case of dynamic groups. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 136–153. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-30574-3_11
6. Bichsel, P., Camenisch, J., Neven, G., Smart, N.P., Warinschi, B.: Get shorty via group signatures without encryption. In: Garay, J.A., De Prisco, R. (eds.) SCN 2010. LNCS, vol. 6280, pp. 381–398. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15317-4_24
7. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_4
8. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28628-8_3
9. Boneh, D., Shacham, H.: Group signatures with verifier-local revocation. In: ACM-CCS 2004, pp. 168–177. ACM Press (2004)
10. Bootle, J., Cerulli, A., Chaidos, P., Ghadafi, E., Groth, J.: Foundations of fully dynamic group signatures. In: Manulis, M., Sadeghi, A.-R., Schneider, S. (eds.) ACNS 2016. LNCS, vol. 9696, pp. 117–136. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39555-5_7
11. Brickell, E.: An efficient protocol for anonymously providing assurance of the container of a private key. In: Submitted to the Trusted Computing Group (2003)
12. Brickell, E., Camenisch, J., Chen, L.: Direct anonymous attestation. In: ACM-CCS 2004, pp. 132–145. ACM Press (2004)
13. Bringer, J., Patey, A.: VLR group signatures - how to achieve both backward unlinkability and efficient revocation checks. In: SECRYPT 2012, pp. 215–220 (2012)
14. Camenisch, J., Lysyanskaya, A.: Dynamic accumulators and application to efficient revocation of anonymous credentials. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 61–76. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_5
15. Chaum, D., van Heyst, E.: Group signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer, Heidelberg (1991). https://doi.org/10.1007/3-540-46416-6_22

16. Chu, C., Liu, J.K., Huang, X., Zhou, J.: Verifier-local revocation group signatures with time-bound keys. In: ASIACCS 2012, pp. 26–27. ACM Press (2012)
17. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0055717
18. Naor, D., Naor, M., Lotspiech, J.: Revocation and tracing schemes for stateless receivers. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 41–62. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_3
19. Emura, K., Hayashi, T.: Road-to-vehicle communications with time-dependent anonymity: a lightweight construction and its experimental results. IEEE Trans. Veh. Technol. **67**, 1582–1597 (2018)
20. Emura, K., Hayashi, T., Ishida, A.: Group signatures with time-bound keys revisited: a new model, an efficient construction, and its implementation. In: ASIACCS 2012, pp. 777–788. ACM Press (2017)
21. Furukawa, J., Imai, H.: An efficient group signature scheme from bilinear maps. In: Boyd, C., González Nieto, J.M. (eds.) ACISP 2005. LNCS, vol. 3574, pp. 455–467. Springer, Heidelberg (2005). https://doi.org/10.1007/11506157_38
22. Gao, T., Peng, F., Guo, N.: Anonymous authentication scheme based on identity-based proxy group signature for wireless mesh network. EURASIP J. Wirel. Commun. Network. **2016**, 193 (2016)
23. Groth, J.: Fully anonymous group signatures without random oracles. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 164–180. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-76900-2_10
24. Ishida, A., Sakai, Y., Emura, K., Hanaoka, G., Tanaka, K.: Fully anonymous group signature with verifier-local revocation. In: Catalano, D., De Prisco, R. (eds.) SCN 2018. LNCS, vol. 11035, pp. 23–42. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98113-0_2
25. Emura, K., Hayashi, T.: A revocable group signature scheme with scalability from simple assumptions and its implementation. In: Chen, L., Manulis, M., Schneider, S. (eds.) ISC 2018. LNCS, vol. 11060, pp. 442–460. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99136-8_24
26. Kilian, J., Petrank, E.: Identity escrow. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 169–185. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0055727
27. Kim, T., Barbulescu, R.: Extended tower number field sieve: a new complexity for the medium prime case. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9814, pp. 543–571. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53018-4_20
28. Kumar, V., Li, H., Park, J., Bian, K., Yang, Y.: Group signatures with probabilistic revocation: a computationally-scalable approach for providing privacy-preserving authentication. In: ACM CCS, pp. 1334–1345 (2015)
29. Libert, B., Ling, S., Nguyen, K., Wang, H.: Zero-knowledge arguments for lattice-based accumulators: logarithmic-size ring signatures and group signatures without trapdoors. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9666, pp. 1–31. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_1
30. Libert, B., Peters, T., Yung, M.: Group signatures with almost-for-free revocation. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 571–589. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_34

31. Libert, B., Peters, T., Yung, M.: Scalable group signatures with revocation. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 609–627. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_36
32. Libert, B., Vergnaud, D.: Group signatures with verifier-local revocation and backward unlinkability in the standard model. In: Garay, J.A., Miyaji, A., Otsuka, A. (eds.) CANS 2009. LNCS, vol. 5888, pp. 498–517. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-10433-6_34
33. Lynn, B.: The pairing-based cryptography library. http://crypto.stanford.edu/pbc/
34. Nakanishi, T., Fujii, H., Hira, Y., Funabiki, N.: Revocable group signature schemes with constant costs for signing and verifying. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 463–480. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00468-1_26
35. Nakanishi, T., Fujiwara, T., Watanabe, H.: A linkable group signature and its application to secret voting. Trans. Inf. Process. Soc. Jpn. **40**(7), 3085–3096 (1999)
36. Nakanishi, T., Funabiki, N.: Verifier-local revocation group signature schemes with backward unlinkability from bilinear maps. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 533–548. Springer, Heidelberg (2005). https://doi.org/10.1007/11593447_29
37. Nakanishi, T., Funabiki, N.: Revocable group signatures with compact revocation list using accumulators. In: Lee, H.-S., Han, D.-G. (eds.) ICISC 2013. LNCS, vol. 8565, pp. 435–451. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-12160-4_26
38. Nguyen, L.: Accumulators from bilinear pairings and applications. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 275–292. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-30574-3_19
39. Ohara, K., Emura, K., Hanaoka, G., Ishida, A., Ohta, K., Saka, Y.: Shortening the libert-peters-yung revocable group signature scheme by using the random oracle methodology. In: IACR Cryptology ePrint Archive, vol. 2016, p. 477 (2016)
40. Perera, M.N.S., Koshiba, T.: Almost-fully secured fully dynamic group signatures with efficient verifier-local revocation and time-bound keys. In: Xiang, Y., Sun, J., Fortino, G., Guerrieri, A., Jung, J.J. (eds.) IDCS 2018. LNCS, vol. 11226, pp. 134–147. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-02738-4_12
41. Rahaman, S., Cheng, L., Yao, D., Li, H., Park, J.: Provably secure anonymousyet-accountable crowdsensing with scalable sublinear revocation. In: PoPETs, vol. 2017, pp. 384–403 (2017)
42. Sadiah, S., Nakanishi, T.: Revocable group signatures with compact revocation list using vector commitments. In: Choi, D., Guilley, S. (eds.) WISA 2016. LNCS, vol. 10144, pp. 245–257. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56549-1_21
43. Sarkar, P., Singh, S.: A general polynomial selection method and new asymptotic complexities for the tower number field sieve algorithm. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016. LNCS, vol. 10031, pp. 37–62. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53887-6_2
44. Yue, X., Chen, B., Wang, X., Duan, Y., Gao, M., He, Y.: An efficient and secure anonymous authentication scheme for vanets based on the framework of group signatures. IEEE Access **6**(1), 62584–62600 (2018)
45. Zhou, S., Lin, D.: Shorter verifier-local revocation group signatures from bilinear maps. In: Pointcheval, D., Mu, Y., Chen, K. (eds.) CANS 2006. LNCS, vol. 4301, pp. 126–143. Springer, Heidelberg (2006). https://doi.org/10.1007/11935070_8