A Guests Managing System with Lattice-based Verifier-local Revocation Group Signature Scheme with Time-bound keys

Maharage Nisansala Sevwandi Perera[1] and Takeshi Koshiba[2]

[1] Graduate School of Science and Engineering
Saitama University, Saitama, Japan
mnisansalasperera@gmail.com,
[2] Faculty of Education and Integrated Arts and Sciences
Waseda University, Tokyo, Japan
tkoshiba@waseda.jp

Abstract. In this paper, we propose a visitors managing system using group signature schemes. The paper focuses on situations that guests who are residing in a hotel are like to keep their activities such as accessing hotel amenities anonymously. Moreover, guests are generally staying in a hotel for a short period. Thus, they should be managed with a proper revocation mechanism. This paper proposes a system using a group signature scheme with Verifier-local revocation and time-bound keys.

Keywords: lattice-based group signatures, almost-full anonymity, dynamical-almost-full anonymity, member registration, verifier-local revocation, time-bound keys

1 Introduction

The group signature schemes, first submitted by Chaum et al. [6] in 1991, allow any member of the group to represent the group without revealing his identity. In this manner, group members are privileged to sign messages for the sake of the group anonymously. Accordingly, signature verifiers can only validate the signatures, but cannot identify the signers. Besides, if necessary, an authority can open the signatures, and recognize the signers. These two features, *anonymity* and *traceability* open doors for practical employment of the group signature schemes. For instance, applications like key-card access systems, digital-right management systems, and e-commerce systems interest using group signature schemes. Most of the first group signature schemes used bilinear map settings. Security of those schemes will be broken once the quantum computers become a reality. Nearly a decade before, in 2010, Gorden et al. [10] put forward the first quantum resist group signature scheme using lattice assumptions. Lattice cryptography is one of the finest auspicious primitive against quantum computers. Lattice cryptography owns provable strong security in reliance on the worst-case hardness of the lattice problems. Moreover lattice cryptography

claims efficient implementation. Thus, Gorden's scheme [10] showed strong security against quantum computers. However, the scheme in [10] has a noticeable disadvantage. The size of the group signature in the scheme in [10] increases with the number of group members N (linear-barrier problem). More specifically, signatures in Gorden's scheme have size $\mathcal{O}(N)$. Later, in 2012, Camenisch et al. [5] offered a more secure and efficient scheme with an anonymous attribute token system. However, this scheme still failed to resolve the linear-barrier problem. Finally, in 2013, Languillaumie et al. [12] suggested a scheme that overcomes the linear-barrier problem. Thus the sizes of the signatures and the group public key in their proposal, given in [12], are proportional to $\log N$. However, none of those above-mentioned group signature schemes from lattices were able to support member registration or revocation. All the three schemes are suitable for static groups, where both the number of group members and their keys are fixed at the time of setup. As a result, no new member can join later, or no existing member can be removed.

Later, in 2014, Langlois et al. [13] suggested the first group signature scheme from lattices that supports member revocation using a revocation method called *Verifier-local Revocation (VLR)*. Thus Langlois's VLR group signature scheme [13] is the first revocation scheme that is accepted to be quantum-resistant at that time. Even though the scheme presented in [13] has several remarkable advantages over the previous works, the security of the scheme is weaker since the scheme depends on a relaxed security notion titled *selfless-anonymity*. The scheme given in [21] offered a security notion known as *almost-full anonymity* for VLR group signature schemes. Even though the almost-full anonymity is stronger than the selfless-anonymity, it shows weaker security than the full-anonymity, suggested by Bellare et al. [3] for static groups. However, the almost-full anonymity can be seen as a reasonable solution for the security of VLR group signature schemes because realizing full-anonymity for VLR group signature schemes is a technically difficult chore. Later, in 2016, Libert et al. [14] formed a group signature scheme from lattice assumptions to facilitate member registration. Their scheme [14] allows new users to register to the group via a group joining protocol. However, even they have considered member registration, they have not focused on facilitating member revocation. Ling et al. [15] offered the first fully dynamic group signature scheme from lattices by using accumulators. Using accumulators seems to be less efficient than using VLR in large groups. Recently, a fully dynamic group signature scheme based on lattices was proposed in [20]. The scheme in [20] serves both user registration and member revocation. The scheme in [20] allows members to join via a joining protocol, and member revocation is managed by using VLR. Moreover, they suggested a new security notion, namely *dynamical-almost-full anonymity*. The dynamical-almost-full anonymity is for fully dynamic (with user registration and member revocation) group signature schemes with VLR and it is an extended version of the almost-full anonymity [21].

Our Contribution

This paper proposes a group signature scheme for managing guest information in a hotel. For handling of guests' check-out, this paper employs Verifier-local Revocation (VLR) technique. The revocation method, VLR utilizes a token system to identify revoked members from active members in a group. Each member in a group has a token, and when a particular member is retiring or removing from the group, the group manager adds the token of that member to a list known as *Revocation List* (*RL*). Then he sends the latest *RL* to the signature verifiers. At the time of verifying, the verifiers scan and check whether signing member's token is not in the list RL. When we implement group signature schemes with VLR to a dynamic system like guest managing system where guests are joining the group temporarily, the length of the revocation list grows in a short time. As a result, the verifiers have to spend a long time to check RL to validate the signer. As a solution to this problem, we can use time-bound keys, which are proposed in [8] by Chu et al. Thus, this paper applies the time-bound keys to the scheme given in [20] and presents a secured and efficient scheme to manage guest information in a hotel.

2 Preliminaries

2.1 Notations

Throughout this paper we express the set of integers $\{1, \ldots, i\}$ by $[i]$ for any integer $i \geq 1$. Matrices are represented in bold uppercase letters such as $\mathbf{X}$, and vectors are in bold lowercase letters, such as $\mathbf{v}$. We consider only column vectors in this paper. While the concatenation of matrices $\mathbf{X} \in \mathbb{R}^{n \times m}$ and $\mathbf{Y} \in \mathbb{R}^{n \times p}$ is presented as $[\mathbf{X}|\mathbf{Y}] \in \mathbb{R}^{n \times (m+p)}$, the concatenation of vectors $\mathbf{v} \in \mathbb{R}^m$ and $\mathbf{z} \in \mathbb{R}^k$ is indicated as $(\mathbf{v}\|\mathbf{z}) \in \mathbb{R}^{m+k}$. If S is a finite set, we mean b is chosen uniformly at random from S by $b \xleftarrow{\$} S$ and if S is a probability distribution we mean b is drawn according to S by $b \xleftarrow{\$} S$. By $\|\mathbf{z}\|$ we denote the Euclidean norm of $\mathbf{z}$. By $\|\mathbf{z}\|_\infty$ we denote the infinity norm of $\mathbf{z}$. χ is a b-bounded distribution over $\mathbb{Z}$. In other words, samples output by χ are with norm at most b with overwhelming probability. Here $b = \sqrt{n}\omega(\log n)$.

2.2 Lattices

Let $\mathbf{V} = [\mathbf{v}_1| \cdots |\mathbf{v}_m] \in \mathbb{Z}_q^{r \times m}$ be linearly independent vectors in $\mathbb{Z}_q^r$ and q be a prime. We define the r-dimensional lattice $\Lambda(\mathbf{V})$ for $\mathbf{V}$ as

$$\Lambda(\mathbf{V}) = \{\mathbf{z} \in \mathbb{Z}^r \mid \mathbf{z} \equiv \mathbf{V}\mathbf{x} \bmod q \text{ for some } \mathbf{x} \in \mathbb{Z}_q^m\}.$$

$\Lambda(\mathbf{V})$ is the set of all linear combinations of columns of $\mathbf{V}$ and m indicates the rank of $\mathbf{V}$.

This paper considers a discrete Gaussian distribution in respect of a lattice. We define the Gaussian function centered in a vector $\mathbf{x}$ with parameter $s > 0$

as $\rho_{s,\mathbf{x}}(\mathbf{z}) = e^{-\pi\|(\mathbf{z}-\mathbf{x})/s\|^2}$ and the corresponding probability density function proportional to $\rho_{s,\mathbf{x}}$ as $D_{s,\mathbf{x}}(\mathbf{z}) = \rho_{s,\mathbf{x}}(\mathbf{z})/s^n$ for all $\mathbf{z} \in \mathbb{R}^n$. For a lattice Λ we specify the discrete Gaussian distribution as $D_{\Lambda,s,\mathbf{x}}(\mathbf{z}) = D_{s,\mathbf{x}}(\mathbf{z})/D_{s,\mathbf{x}}(\Lambda) = \rho_{s,\mathbf{x}}(\mathbf{z})/\rho_{s,\mathbf{x}}(\Lambda)$ for all $\mathbf{z} \in \Lambda$. As $\mathbb{Z}^m$ is also a lattice, we can declare a discrete Gaussian distribution for $\mathbb{Z}^m$. Thus, by $D_{\mathbb{Z}^m,\sigma}$, we express the discrete Gaussian distribution for $\mathbb{Z}^m$ around the origin with the standard deviation σ.

2.3 Lattice-Related Hardness Problems

Learning With Errors (LWE)

Definition 1. *Learning With Errors (LWE) [18]: Let $n, m \geq 1$, and $q \geq 2$ be integers. For $\boldsymbol{s} \in \mathbb{Z}_q^n$ and χ, let the distribution $\mathrm{A}_{\boldsymbol{s},\chi}$ gained by sampling uniformly random $\boldsymbol{a} \in \mathbb{Z}_q^n$ and selecting $\mathrm{e} \leftarrow \chi$, and resulting the pair $(\boldsymbol{a}, \boldsymbol{a}^T \cdot \boldsymbol{s} + \mathrm{e})$.*

Search-LWE and Decision-LWE are the two versions of LWE problems. For a given LWE samples Search-LWE is determined to find the secret $\mathbf{s}$. Decision-LWE is for distinguishing LWE samples and samples selected according to the uniformly distribution. This paper uses the hardness of the problem Decision-LWE.

We say, for a prime power q, $b \geq \sqrt{n}\omega(\log n)$, and distribution χ, solving $LWE_{n,q,\chi}$ problem is at least as hard as solving $SIVP_\gamma$, where $SIVP$ means *Shortest Independent Vector Problem* and $\gamma = \tilde{\mathcal{O}}(nq/b)$ [9, 22].

Short Integer Solution ($\boldsymbol{SIS_{n,m,q,\beta}}$) In 1996, SIS was first explained in seminal work of Ajtai [2]. SIS problem is for finding a sufficiently short nontrivial integer combination of given uniformly random elements of a certain large finite additive group, which sums to zero [18].

Definition 2. *Short Integer Solution ($SIS_{n,m,q,\beta}$ [18, 22]): For given m uniformly random vectors $\boldsymbol{a}_i \in \mathbb{Z}_q^n$, which forms the columns of a matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$, SIS requires to find the vector $\boldsymbol{z} \in \mathbb{Z}^m$, which is a nonzero vector and satisfies $\|\boldsymbol{z}\| \leq \beta$ and $\boldsymbol{A}\boldsymbol{z} = 0 \mod q$.*

We say, for any m, β, and for any $q \leq \sqrt{n\beta}$, solving $SIS_{n,m,q,\beta}$ problem with non-negligible probability is at least as hard as solving $SIVP_\gamma$ problem, for some $\gamma = \beta \cdot \mathrm{O}(\sqrt{n})$ [9].

2.4 Lattice Algorithms

To construct the propose scheme, this paper uses algorithms $\mathsf{SampleD}$ [9, 16] and $\mathsf{GenTrap}$ [1, 9, 16]. The algorithm $\mathsf{SampleD}$ is a randomized nearest-plane algorithm and it samples from a discrete Gaussian $D_{\Lambda,s,\mathbf{c}}$ over any lattice Λ. $\mathsf{GenTrap}$ is a preimage sampleable trapdoor function (PSTF).

- $\mathsf{SampleD}(\mathbf{T_A}, \mathbf{A}, \mathbf{u}, \sigma)$: On inputs a vector $\mathbf{u}$ in the image of $\mathbf{A}$, a trapdoor $\mathbf{T_A}$, and $\sigma = \omega(\sqrt{n \log q \log n})$, $\mathsf{SampleD}$ results $\mathbf{z} \in \mathbb{Z}^m$ sampled from the distribution $D_{\mathbb{Z}^m,\sigma}$. The output $\mathbf{z}$ should assure $\mathbf{A} \cdot \mathbf{z} = \mathbf{u} \bmod q$.

– $\mathsf{GenTrap}(n,\ m,\ q)$: $\mathsf{GenTrap}(n,\ m,\ q)$ results a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a trapdoor $\mathbf{T_A}$ for any given integers $n \geq 1, q \geq 2$, and sufficiently large $m = O(n \log q)$. The distribution of the resulted $\mathbf{A}$ is negl(n)-far from the uniform distribution.

2.5 One-Time Signature Scheme

Our new scheme uses one-time signature scheme $\mathcal{OTS} = (\mathsf{OGen}, \mathsf{OSign}, \mathsf{OVer})$ [17]. $\mathcal{OTS}$ schemes are digital signature schemes based on one-way functions. While OGen is an algorithm for generating keys, OSign and OVer are algorithms for producing signatures and verifying signatures. OGen takes (1^n), and creates a signing, verification key pair (**osk**, **ovk**). For **osk** and a message M, OSign produces a signature Σ. OVer takes as inputs **ovk**, a message M, and a signature Σ and outputs $\top$ or $\bot$ [7].

3 Applying VLR Group Signature Schemes with Time-bound keys to the Guest-Managing System

This section first discusses the primitives used to construct the new VLR group signature scheme with time-bound keys and then explains the application of the guest-managing system. Thus, it defines the general VLR group signature scheme and the dynamical-almost full anonymity with the scheme proposed in [20]. Next, it describes time-bound keys. Finally, it discusses the application of time-bound keys and VLR scheme on guest managing system.

3.1 VLR group signature schemes

Even though general group signature schemes have four algorithms, KeyGen, Sign, Verify, and Open, group signature schemes with VLR have of only the first three algorithms. Since VLR group signature schemes have an *implicit tracing algorithm* which requires to execute Verify for each member until the signer is revealed, those schemes do not need the algorithm Open for tracing signers.

– $\mathsf{KeyGen}(n,N)$: KeyGen is a randomized PPT algorithm, and on input of n and N it produces a group public key **gpk**, a set of group members secret keys $\mathbf{gsk} = (\mathbf{gsk}[0], \ldots, \mathbf{gsk}[N-1])$, and a set of group members revocation tokens $\mathbf{grt} = (\mathbf{grt}[0], \ldots, \mathbf{grt}[N-1])$. While $n \in \mathbb{N}$ is the security parameter, N is the number of group users.
– $\mathsf{Sign}(\mathbf{gpk},\ \mathbf{gsk}[d],\ M)$: Sign is a randomized algorithm that creates a group signature Σ on a given message M. Sign takes secret signing key $\mathbf{gsk}[d]$, the group public key **gpk**, and a message $M \in \{0,1\}^*$ as inputs.
– $\mathsf{Verify}(\mathbf{gpk},\ RL,\ \Sigma,\ M)$: Verify is a deterministic algorithms. With the given group public key **gpk** Verify checks if the input signature Σ is valid for the input message M and validates the signer not being revoked using the latest RL.

3.2 Dynamical-almost-full anonymity

In the full -anonymity game [3] between an adversary and a challenger, in the beginning, the challenger gives all the secret signing keys of the members to the adversary. However, for VLR group signature schemes we cannot achieve the full-anonymity because VLR group signatures have tokens and tokens cannot be given to the adversary. If any token is gained by the adversary, he can execute the algorithm Verify with the token he has and can identify whether the owner of the token generated the signature or not. In previous VLR group signature schemes [13], tokens for the members are formed using a part of the secret signing keys of the members. Thus, we cannot allow the adversary to get any secret signing keys also because he can obtain the tokens from the secret signing keys. Hence, VLR group signatures rely on the selfless-anonymity which restricts revealing any information related to the challenging signature of the adversary.

The scheme for the VLR group signatures with both member registration and revocation, given in [20], provides stronger security than the selfless-anonymity called *dynamical-almost-full anonymity*. In the beginning of the anonymity game of the dynamical-almost-full anonymity, the challenger gives all the present members' secret signing keys to the adversary. Thus, the dynamic-almost-full anonymity is applicable only for the schemes where the tokens are generated separately to the secret signing keys. Then the anonymity game of the dynamical-almost-full anonymity enables the adversary insert new members. However, at the time of adversary registering as a new user, we do not provide his revocation token to him. The adversary can request revocation token of any member later. At the challenging phase, we generate the challenging signature only for the noncorrupted members added by the adversary. Thus we do not create a challenging signature for members whose token is revealed to the adversary and who is not added by the adversary as new members.

3.3 Time-bound keys

When members are joining a group temporarily, the size of the revocation list (RL) increases quickly. Thus the verifiers have to check whether the signer of a given signature is valid by scanning a long list. Checking a long list decreases the efficiency of Verify. Chu et al. [8] suggested a solution called time-bound keys for reducing the cost of the revocation check. In Chu's scheme, the group members' keys and the signatures have expiration dates. At the time of adding a new user to the group, the group manager should determine an expiration date for the new member keys. Thus every registered member's secret key has an expiration date, and only the members having non-expired keys are allowed to generate signatures. This cuts down the cost of checking the expired members' (naturally revoked members) revocation statue at the signature verification. Since expired members are not allowed to generate valid signatures, the group manager does not need to add their revoked details to the revocation list RL.

To efficiently compare two dates, Chu et al. [8] have used the date format as "YYMMDD" in integer form. For instance, the date 2019 December 24th is

indicated as "191224". Moreover, $t_1 > t_2$ represents that the date t_1 is later than the date t_2. In Chu's scheme, the key expiration date, which is selected by the group manager, is represented by t_r. Moreover, at the signature generation, the signer is allowed to choose a signature expiration date t_s which should satisfy $t_r > t_s$ and $t_s \geq t_v$, where t_v is the verification date. A valid signature should satisfy $t_r > t_s \geq t_v$.

3.4 VLR group signature scheme with time-bound keys for a Guest-managing system

When a guest G_i proceeds the hotel check-in, G_i provides check-in information including information regarding his stay. The receptionist or the hotel manager or the system issues G_i a key-card with some details that he is allowed to access. The key-card only can be used until the leaving date. This restriction can be applied using time-bound keys. Thus the key-card have an expiration date (t_{r_i}). The guest G_i can utilize the hotel amenities (bar, pool, spa, and gym) using the key-card until he leaves. However, the key-card will not reveal any personal information like name, room number of the guest. The system can only verify the key-card is not expired and is valid to use the facility. All the amenities have a limited period of utilizing them. For instance, the pool can be used only for two hours at a time. To control the period of using amenities, again the time-bound keys can be used. Each time the guest is accessing an amenity can be regarded as issuing a signature. Thus, the expiration time of using the pool is the expiration time of the signature (t_s).

We use the date format as "YYYYMMDDhhmm". If G_i's check-out date and time is 2018 August 09 morning 10, then his key-card expiration time (t_r) is 201808091000. If he accesses the gym in 2018 August 07 evening 4:00 which is allowed only for two hours, then his signature expiration time will be 201808071800. After finishing the work out in the gym, he has to enter his card to go out of the gym. At that time system will validate his signature expiration time with the leaving time. If he has used the gym for more than two hours, then he has to pay the penalty. If he used an expired key card, then he will not be able even to enter the gym. When accessing the gym, the key-card generates a signature with signature expire time (t_s), and the system validates that the signature expiration time is greater than or equal to the current time (t_v). When he leaves, the system again checks that the signature expiration time is greater than or equal to the current time (t_v). For instance, as discussed before, the guest enters the gym at 4 pm on 2018 August 07, and the gym allows only two hours of work-out. Then t_s is 201808071800 and entering time (t_v) is 201808071600. Since $t_s > t_v$ and $t_r > t_s$ he is allowed to enter. If he finishes the workout at 5:50 pm then the system allows him to exit without any issue since the leaving time (201808071750) $t_v < t_s$. But if he tries to leave at 6:30 pm (201808071830), then system asks him to pay additional charge because $t_v > t_s$ (he used the gym extra 30 minutes).

Using the time-bound keys, we can manage the use of amenities and the room in a hotel guest-managing system as discussed.

4 New Scheme

This section first, defines the underlying interactive protocol that the propose scheme uses to confirm the validity of the signers. Then this section provides the description of the new scheme.

4.1 Supporting zero-knowledge protocol

In this section, we briefly explain the zero-knowledge proof system that our scheme uses as underlying interactive protocol. Let COM be the statistically hiding and computationally binding commitment scheme [11]. In our scheme, matrices $\mathbf{F}$, $\mathbf{A}$, $\mathbf{B}$, $\mathbf{V}$, $\mathbf{G}$, $\mathbf{H}$ and vectors $\mathbf{u}$, $\mathbf{v}$, $\mathbf{c}_1$, $\mathbf{c}_2$ are the public parameters. The witness of the prover consists of vectors $\mathbf{x}$, $\mathsf{bin}(\mathbf{z})$, $\mathbf{r}$, $\mathbf{s}$, $\mathbf{e}_1$, and $\mathbf{e}_2$. The goal of the prover is to prove $\mathbf{F} \cdot \mathbf{x} = \mathbf{H}_{4n\times 2m} \cdot \mathsf{bin}(\mathbf{z})$ (as given in [14]) and $\mathbf{V} \cdot (\mathbf{A} \cdot \mathbf{r}) + \mathbf{e}_1 = \mathbf{v} \mod q$ and $(\mathbf{c}_1 = \mathbf{B}^T\mathbf{s} + \mathbf{e}_1, \mathbf{c}_2 = \mathbf{G}^T\mathbf{s} + \mathbf{e}_2 + \lfloor q/2 \rfloor \mathsf{bin}(\mathbf{z}_i))$ (as discussed in [19]) to the verifier. Here $\mathbf{H}_{n\times n\lceil \log q\rceil} \in \mathbb{Z}^{n\times n\lceil \log q\rceil}$ is a "power-of-2" matrix and $\mathbf{z} = \mathbf{H}_{n\times n\lceil \log q\rceil} \cdot \mathsf{bin}(\mathbf{z})$ for any $\mathbf{z} \in \mathbb{Z}_q^n$.

4.2 Description of the new Scheme

The new scheme has algorithms, KeyGen, Join, Sign, Verify, Open, and Revoke. The proposed scheme is constructed based on the scheme given in [20]. Using the time-bound keys the methods in the scheme given in [20] are modified and the new scheme is presented. Thus, the algorithms KeyGen (setup), Open, and Revoke are same as the algorithms given in [20]. The algorithms Join, Sign, and Verify are modified by adding the generation and validation of the time-bound keys.

The security parameter is denoted by λ. We denote the maximum number of predicted members in a group by $N = 2^\ell$. Then we choose lattice parameter $n = \mathcal{O}(\lambda)$, prime modulus $q = \tilde{\mathcal{O}}(\ell n^3)$, Gaussian parameter $\sigma = \Omega(\sqrt{n \log q} \log n)$, dimension $m = 2n\lceil \log q \rceil$, infinity norm bounds $\beta = \sigma\omega(\log m)$ and $b = \sqrt{n}\omega(\log n)$.

The algorithms of the new scheme are as follows.

Setup: The randomized algorithm $\mathsf{KeyGen}(1^n, 1^N)$ works as follows.

1. Obtain $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a trapdoor $\mathbf{T_A}$ by executing the PPT algorithm $\mathsf{GenTrap}(n,\ m,\ q)$.
2. Sample vector $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^n$.
3. Execute $\mathsf{GenTrap}(n,\ m,\ q)$ to obtain the encryption and decryption keys $\mathbf{B} \in \mathbb{Z}_q^{n\times m}$ and a trapdoor $\mathbf{T_B}$.
4. Select a matrix $\mathbf{F} \xleftarrow{\$} \mathbb{Z}_q^{4n\times 4m}$.
5. Finally, output ($\mathbf{A}$, $\mathbf{B}$, $\mathbf{F}$, $\mathbf{u}$) as the group public key **gpk**, the group manager's (issuer/ issuing manager) secret key **ik**:= $\mathbf{T_A}$, and the tracing manager's (opener) secret key **ok**:= $\mathbf{T_B}$.

Join: A guest (new user) i having a personal public key and private key pair $(\mathbf{upk}[i], \mathbf{usk}[i])$ communicate with the group manager to join the group by following the steps below.

1. User i, the new user (guest), samples a discrete Gaussian vector $\mathbf{x}_i \leftarrow D_{\mathbb{Z}^{4m},\sigma}$, and computes $\mathbf{z}_i \leftarrow \mathbf{F} \cdot \mathbf{x}_i \in \mathbb{Z}_q^{4n}$. Then he makes a signature $\Sigma_{join} \leftarrow \mathsf{Sig}(\mathbf{usk}[i], \mathbf{z}_i)$ and passes $\mathbf{z}_i$, and Σ_{join} to the group manager. Here, since the user (guest) has a leaving date (check-out date and time) he sends that date and time also along with $\mathbf{z}_i$ and Σ_{join}. Thus the user (guest) i sends the group manager $\mathbf{z}_i$, Σ_{join} , and t_{r_i}.
2. The group manager confirms that $\mathbf{z}_i$ was not owned by any previous member and t_{r_i} is correct and later than the current date. Then he validates Σ_{join} is a valid signature created on $\mathbf{z}_i$, using $\mathsf{Vf}(\mathbf{upk}[i], \mathbf{z}_i, \Sigma_{join})$. He terminates if $\mathbf{z}_i$, and Σ_{join} is invalid. Otherwise he signs the user's index $d = \mathsf{bin}(\mathbf{z}_i)$ and t_{r_i}, using his secret key $\mathsf{bin}(\mathbf{z}_i)$ is the binary representation of $\mathbf{z}_i$. Then he generates the certificate for the index and the expiration date (check-out date) $\textit{cert-index}_i = \mathsf{Sign}(\mathbf{ik}, (\mathsf{bin}(\mathbf{z}_i), t_{r_i}))$. Thus, later the user (guest) i cannot fake his check-out date and time.
   The group manager selects $\mathbf{R}_i \xleftarrow{\$} \mathbb{Z}_q^{n\times 4n}$ and computes $\mathbf{w}_i = \mathbf{R}_i \cdot \mathbf{z}_i$. Then he samples a vector $\mathbf{r}_i \in \mathbb{Z}^m \leftarrow \mathsf{SampleD}(\mathbf{T_A}, \mathbf{A}, \mathbf{u} - \mathbf{w}_i, \sigma)$, and creates a certificate for the token $\textit{cert-token}_i = \mathsf{Sign}(\mathbf{ik}, (\mathbf{A} \cdot \mathbf{r}_i))$ $(\mathbf{ik} = \mathbf{T_A})$.
   Next in the registration table, he records the information of the new user i as $\textit{reg}[i] \leftarrow (i, d, \mathbf{upk}[i], \mathbf{z}_i, \Sigma_{join}, \mathbf{R}_i, \mathbf{w}_i, t_{r_i}, \mathbf{r}_i, 1)$ and makes the record active (1).
   Finally, the group manager transfers the new member's member-certificate $\textit{cert}_i = (\textit{cert-index}_i, \textit{cert-token}_i, \mathbf{R}_i, (\mathbf{A} \cdot \mathbf{r}_i), t_{r_i})$.

Sign: $\mathsf{Sign}(\mathbf{gpk}, \mathbf{gsk}[i], \textit{cert}_i, M, t_s)$ is a randomized algorithm and on the given message M it produces a signature Σ employing the signer (guest) secret key $\mathbf{gsk}[i] = \mathbf{x}_i$ as follows.

1. Let $\mathcal{H}_1$: $\{0,1\}^* \rightarrow \mathbb{Z}_q^{n\times \ell}$, $\mathcal{H}_2$: $\{0,1\}^* \rightarrow \{1,2,3\}^t$ and $\mathcal{G}$: $\{0,1\}^* \rightarrow \mathbb{Z}_q^{n\times m}$. $\mathcal{H}_1$, $\mathcal{H}_2$, and $\mathcal{G}$ are hash functions modeled as a random oracle.
2. Parse $\mathbf{gpk}$ as $(\mathbf{A, B, F, u})$.
3. Parse $\textit{cert}_i$ as $(\textit{cert-index}_i, \textit{cert-token}_i, \mathbf{R}_i, (\mathbf{A} \cdot \mathbf{r}_i), t_{r_i})$.
4. If $t_s > t_{r_i}$ then return ε.
5. Run $\mathsf{OGen}(1^n) \rightarrow (\mathbf{ovk}, \mathbf{osk})$.
6. Encrypt the index $d = \mathsf{bin}(\mathbf{z}_i)$, where $\mathbf{z}_i = \mathbf{F} \cdot \mathbf{x}_i$.
   (a) Let $\mathbf{G} = \mathcal{H}_1(\mathbf{ovk}) \in \mathbb{Z}_q^{n\times 2m}$.
   (b) Sample $\mathbf{s} \leftarrow \chi^n$, $\mathbf{e}_1 \leftarrow \chi^m$ and $\mathbf{e}_2 \leftarrow \chi^\ell$.
   (c) Compute the ciphertext $(\mathbf{c}_1, \mathbf{c}_2)$ pair $(\mathbf{c}_1 = \mathbf{B}^T\mathbf{s} + \mathbf{e}_1,\ \mathbf{c}_2 = \mathbf{G}^T\mathbf{s} + \mathbf{e}_2 + \lfloor q/2 \rfloor \mathsf{bin}(\mathbf{z}_i))$.
7. Select $\rho \xleftarrow{\$} \{0,1\}^n$, let $\mathbf{V} = \mathcal{G}(\mathbf{A}, \mathbf{u}, M, \rho) \in \mathbb{Z}_q^{n\times m}$.
8. Get $\mathbf{v} = \mathbf{V}\cdot(\mathbf{A}\cdot\mathbf{r}_i)+\mathbf{e}_1 \mod q$ ($||\mathbf{e}_1||_\infty \leq \beta$ with overwhelming probability).

9. Execute $\mathsf{Verify}(\mathbf{A}, (\mathsf{bin}(\mathbf{z}_i), t_{r_i}), \textit{cert-index}_i)$ to prove that $\textit{cert-index}_i$ is generated on $(\mathsf{bin}(\mathbf{z}_i), t_{r_i})$ and $\mathsf{Verify}(\mathbf{A}, (\mathbf{A} \cdot \mathbf{r}_i), \textit{cert-token}_i)$ to prove that $\textit{cert-token}_i$ is created on $(\mathbf{A} \cdot \mathbf{r}_i)$. Next make a proof as in Section 4.1, to show the signer is valid, above $\mathbf{v}$ is sincerely computed, and index is correctly encrypted. Then repeat the protocol given in Section 4.1 $t = \omega(\log n)$ times to make the soundness error of the interactive protocol negligible. Finally, make it non-interactive using the Fiat-Shamir heuristic as a triple, $\Pi = (\{CMT^{(k)}\}_{k=1}^{t}, CH, \{RSP^{(k)}\}_{k=1}^{t})$, where CH $= (\{Ch^{(k)}\}_{k=1}^{t}) = \mathcal{H}_2(M, \{CMT^{(k)}\}_{k=1}^{t}, \mathbf{c}_1, \mathbf{c}_2)$.
10. Compute $\mathcal{OTS}$; $sig = \mathsf{OSig}(\mathbf{osk}, (\mathbf{c}_1, \mathbf{c}_2, \Pi))$.
11. Produce the signature $\Sigma = (\mathbf{ovk}, (\mathbf{c}_1, \mathbf{c}_2), \rho, \Pi, sig, \mathbf{v}, t_{r_i}, t_s)$.

Verify: $\mathsf{Verify}(\mathbf{gpk}, M, \Sigma, RL, t_v)$ is a deterministic algorithm and it proceeds as follows, where $RL = \{\{\mathbf{u}_i\}_i\}$ and t_v is the current time of validation.

1. Parse Σ as $(\mathbf{ovk}, (\mathbf{c}_1, \mathbf{c}_2), \rho, \Pi, sig, \mathbf{v}, t_{r_i}, t_s)$.
2. If $t_v > t_s$ or $t_s > t_{r_i}$ then return 0.
3. Get $\mathbf{V} = \mathcal{G}(\mathbf{A}, \mathbf{u}, M, \rho) \in \mathbb{Z}_q^{n \times m}$.
4. If $\mathsf{OVer}(\mathbf{ovk}, ((\mathbf{c}_1, \mathbf{c}_2), \Pi), sig) = 0$ then output 0.
5. Parse Π as $(\{CMT^{(k)}\}_{k=1}^{t}, \{Ch^{(k)}\}_{k=1}^{t}, \{RSP^{(k)}\}_{k=1}^{t})$.
6. If $(Ch^{(1)}, \ldots, Ch^{(t)}) \neq \mathcal{H}_2(M, \{CMT^{(k)}\}_{k=1}^{t}, \mathbf{c}_1, \mathbf{c}_2)$ return 0 else continue.
7. Using the verification steps of the commitment scheme in Section 4.1 for $k = 1$ to t validate $RSP^{(k)}$ with respect to $CMT^{(k)}$ and $Ch^{(k)}$. Return invalid if any of the conditions fails.
8. Compute $\mathbf{e}_i^{'} = \mathbf{v} - \mathbf{V} \cdot \mathbf{u}_i \mod q$ for each $\mathbf{u}_i \in RL$ to certify whether there exists an index i such that $\|\mathbf{e}_i^{'}\|_\infty \leq \beta$. If so output invalid and abort.
9. Output valid.

Open: $\mathsf{Open}(\mathbf{gpk}, \mathbf{ok}, \textit{reg}, M, \Sigma)$ acts as bellow. Here $\mathbf{ok} = \mathbf{T_B}$.

1. Let $\mathbf{G} = \mathcal{H}_1(\mathbf{ovk})$.
2. Use $\mathbf{T_B}$ to obtain $\mathbf{Y} \in \mathbb{Z}^{m \times 2m}$. Here $\mathbf{Y}$ is a small norm matrix and $\mathbf{B} \cdot \mathbf{Y} = \mathbf{G} \mod q$.
3. Compute $\mathsf{bin}(\mathbf{z}_i) = \lfloor (\mathbf{c}_2 - \mathbf{Y}^T \cdot \mathbf{c}_1)/(q/2) \rceil$, identify the signer querying the registration table *reg* with $\mathsf{bin}(\mathbf{z}_i)$, and return the signer's index.

Revoke: $\mathsf{Revoke}(\mathbf{gpk}, \mathbf{ik}, i, \textit{reg}, \mathrm{RL})$ works as below.

1. Query *reg* for i to get the revocation token $(\mathbf{A} \cdot \mathbf{r}_i)$ of the revoking member.
2. Insert $(\mathbf{A} \cdot \mathbf{r}_i)$ to RL and modify $\textit{reg}[i]$ to inactive (0).
3. Output RL.

5 Correctness and Security Analysis of the Scheme

This section provides an analysis of the correctness and the security of the scheme.

5.1 Correctness

Theorem 1. *For all* $t_r, t_s, t_v, RL, M \in \{0,1\}^*, (\boldsymbol{gpk}, \boldsymbol{ok}, \boldsymbol{ik}) \leftarrow \mathsf{GKg}(1^\lambda)$ *and* $(\boldsymbol{gsk}[i], \boldsymbol{grt}[i], cert_i) \leftarrow \mathsf{Join}$,

$\mathsf{Verify}(\boldsymbol{gpk}, M, \mathsf{Sign}(\boldsymbol{gpk}, \boldsymbol{gsk}[i], \boldsymbol{grt}[i], M, t_s), RL, t_v) = \mathsf{valid} \iff \boldsymbol{grt}[i] \notin RL$ *and* $t_r > t_s \geq t_v$ *and*

$\mathsf{Open}(\boldsymbol{gpk}, \boldsymbol{ok}, reg, M, \mathsf{Sign}(\boldsymbol{gpk}, \boldsymbol{gsk}[i], \boldsymbol{grt}[i], M, t_s)) = i.$

The new scheme allows only new users whose public keys are not used before to join the group. The group manager validates the new user and issues a certificate with the key-expiration time (guest's check-out). Thus any user cannot pass the signature generation with a fake expiration date (t_r). Moreover, at the time of signature generation, first, the key-expiration is checked. If the signer cannot provide a valid signature expiration date, then he cannot proceed with generating signatures. This confirms that the no expired member can generate a signature. Next the signatures which are only produced by active and honest users with a valid signature expiration date are accepted by Verify. If the provided signature is expired, then the signature verification fails. Further, signatures created by users whose token is in RL are not accepted by Verify. Both Sign and Verify ensures this condition. Completeness of the underlying proof system ensures to accept legitimate signatures always. The soundness of the underlying proof system assure that revoked signers fail get acceptance for their signatures. Open returns the index of the signature owner with overwhelming probability. Open computes $\mathsf{bin}(\mathbf{z}_i)$, and get the information of the signer from *reg*.

Above discussion proves the correctness of the new scheme.

5.2 Anonymity (dynamical-almost-full anonymity)

Theorem 1: *Under the hardness of* $LWE_{n,q,\chi}$ *problem the new scheme is dynamical-almost-full anonymous in the random oracle mode.*

With the following sequence of games we prove the anonymity of the proposed scheme.

Game 0: First, the challenger C obtains a group public key **gpk** and other keys from $\mathsf{KeyGen}(1^n, 1^N)$. Next, C delivers **gpk** and all the group members' secret keys **gsk** to the adversary$A$. In the query phase, A can access opening for any desired signature and he can ask for any member revocation tokens. Furthermore, A can add new users to the group. When A requests to add a new user, C validates and adds the new user details to the registration table *reg* and a list called **HU**. But C will not give the revocation token of the new user. Thus, even the member certificate including the token and the key-expiration date and time is generated and saved, the member certification will not be given in the registration query. C gives a success message only for member registration of the adversary. Moreover, when A requests to reveal any user token, C traces the requested user's index by adding the request to a list called **TU**, and returns the member certificate *cert* which was saved in the registration table at the

time of registering. In the challenge phase, A sends two indices (i_0, i_1) with a message M^*. Only if (i_0, i_1) are newly added (in **HU**) and are not used for querying tokens (not in **TU**), then C makes and responds back a signature $\Sigma^* = (\mathbf{ovk}^*, (\mathbf{c}_1^*, \mathbf{c}_2^*), \rho^*, \Pi^*, sig^*, \mathbf{v}^*, t_{r_i}^*, t_s^*) \leftarrow \mathsf{Sign}(\mathbf{gpk}, \mathbf{gsk}[i_b]^*, cert_{i_b}, M^*, t_s^*)$ for a random $b \leftarrow \{0,1\}$. Finally, A provides $b' \in \{0,1\}$ as the guess of b. If $b' = b$ then outputs 1 or 0 otherwise.

The adversary may try to win the game by sending the challenging indices with two different expiration dates. However, he cannot use this trick to win the game as the challenger will generate the challenging signature only if both indices having valid expiration dates. Both indices should satisfy $t_r > t_s$. Moreover, it should satisfy $t_r > t_s \geq t_v$. Thus the adversary cannot provide any indices with key-expiration dates that fail only at the time of validating and that passes signature generation. Thus, the adversary cannot attack the anonymity of the scheme using the time-bound keys.

Game 1: In this game, C makes a trivial amendment comparing to the above game **Game** 0. The challenger C creates $\mathcal{OTS}$ key pair $(\mathbf{ovk}^*, \mathbf{osk}^*)$ at the start of the game which is generated at the time of signature generation in general. If A request opening of a valid signature $\Sigma = (\mathbf{ovk}, (\mathbf{c}_1, \mathbf{c}_2), \rho, \Pi, sig, \mathbf{v}, t_r, t_s)$, where $\mathbf{ovk}$=$\mathbf{ovk}^*$, then C aborts the game by providing a random bit. Besides, $\mathbf{ovk}$=$\mathbf{ovk}^*$ contradicts the strong unforgeability of $\mathcal{OTS}$. On the other hand, as adversary cannot know $\mathbf{ovk}^*$, $\mathbf{ovk}$=$\mathbf{ovk}^*$ has negligible probability. Moreover, if A provides a valid signature, which satisfies $\mathbf{ovk}$=$\mathbf{ovk}^*$, then sig is a forged signature. We believe that A does not ask for opening of a valid signature with $\mathbf{ovk}^*$. As a result, C halting the game is negligible.

Game 2: In this game, C creates the random oracle $\mathcal{H}_1$ at the start, which is generally obtained at the signature generation. In this game, at the beginning, C substitutes $\mathbf{B}$ and $\mathbf{G}$, which are the encrypting matrices. C picks $\mathbf{B}^* \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{G}^* \in \mathbb{Z}_q^{n \times \ell}$ in uniformly random. Then he sets $\mathcal{H}_1(\mathbf{ovk}^*)$=$\mathbf{G}^*$. To response the opening oracle queries of A with $\Sigma = (\mathbf{ovk}, (\mathbf{c}_1, \mathbf{c}_2), \rho, \Pi, sig, \mathbf{v}, t_r, t_s)$, C chooses $\mathbf{Y} \leftarrow (D_{z^m,\sigma})^\ell$, and computes $\mathbf{G} = \mathbf{B}^*\mathbf{Y} \in \mathbb{Z}_q^{n \times \ell}$. This $\mathbf{G}$ is used to response the opening and records ($\mathbf{ovk}$, $\mathbf{Y}$, $\mathbf{G}$) to be reused if the adversary A repeats the same queries for $\mathcal{H}_1(\mathbf{ovk})$. Since the distributions of $\mathbf{G}$ is statistically close to the uniform over $\mathbb{Z}_q^{n \times \ell}$ [9], this game is indistinguishable from Game 1.

Game 3: In this game, without genuinely producing the non-interactive proof Π, without using the witness C simulates the proof. For this he executes the simulator for each $k \in [t]$ and then programs the random oracle $\mathcal{H}_1$ respectively. The challenging signature Σ^* is statistically close to the signature in the previous games as the proof system is statistically zero-knowledge. Thus, Game 3 is indistinguishable from previous game.

Game 4: C substitutes the naive revocation token with a vector sampled uniformly. In general, $\mathbf{v} = \mathbf{V} \cdot \mathbf{grt}[i_b] + \mathbf{e}_1 \mod q$. But in this game, C uniformly samples $\mathbf{t} \xleftarrow{\$} \mathbb{Z}_q^n$ and get $\mathbf{v} = \mathbf{V} \cdot \mathbf{t} + \mathbf{e}_1 \mod q$. $\mathbf{V}$ is uniformly random over $\mathbb{Z}_q^{m \times n}$ and $\mathbf{e}_1$ is sampled from the error distribution χ. Since C substitutes only $\mathbf{grt}[i_b]$ with $\mathbf{t}$ and the rest of the game is same as previous game, the two games are statistically indistinguishable.

Game 5: In this game, $\mathbf{v}$ is taken uniformly. In this game, C makes the revocation token details without depending on the challenging bit b. Accordingly, C gets $\mathbf{y} \xleftarrow{\$} \mathbb{Z}_q^m$. Then he sets $\mathbf{v} = \mathbf{y}$. In **Game** 4, the pair $(\mathbf{V}, \mathbf{v})$ is a proper $LWE_{n,q,\chi}$ instance. In this game C substitutes $\mathbf{v}$ with $\mathbf{y} \xleftarrow{\$} \mathbb{Z}_q^m$ which is sampled uniformly. The adversary can solve Decision-LWE problem, if he can distinguish $\mathbf{v}$ and $\mathbf{y}$. Since the lattice problem $LWE_{n,q,\chi}$ is hard, Game 5 is indistinguishable from Game 4.

Game 6: In this game C amends the making of ciphertext $(\mathbf{c}_1^*, \mathbf{c}_2^*)$ uniformly. He sets $\mathbf{c}_1^* = \mathbf{x}_1$ and $\mathbf{c}_2^* = \mathbf{x}_2 + \lfloor q/2 \rfloor d_b$. The vectors $\mathbf{x}_1 \in \mathbb{Z}^m$ and $\mathbf{x}_2 \in \mathbb{Z}^\ell$ are uniformly random. d_b is the index of the challenging bit. The rest of the game is same as Game 5. Since $LWE_{n,q,\chi}$ is hard, Game 5 and Game 6 are indistinguishable.

Game 7: C creates Σ^* totally independent of the bit b. He picks uniformly random $\mathbf{x}_1' \in \mathbb{Z}_q^m$ and $\mathbf{x}_2' \in \mathbb{Z}_q^\ell$, and sets $\mathbf{c}_1^* = \mathbf{x}_1'$ and $\mathbf{c}_2^* = \mathbf{x}_2'$. Game 7 is statistically indistinguishable from Game 6. The advantage of the adversary in this game is 0 because the signature is totally independent from the challenger's bit b.

Hence, these games prove that the new scheme with the time -bound keys is secure with the dynamical-almost-full anonymity.

5.3 Traceability

Theorem 2: *Under the hardness of **SIS** problem, the new scheme is traceable in the random oracle model.*

We take an algorithm B that capable of solving SIS problem with non-negligible probability. An adversary A having **gpk** and **ok** returns (M, Σ) in the traceability game. Moreover the adversary A has facility to add new users and replace members' personal public keys. Also, A is allowed to query any member secret signing key and revocation token. By using oracles, B answers the queries done by A.

Finally, the adversary A outputs a fake signature $\Sigma^*=(\mathbf{ovk}^*, (\mathbf{c}_1^*, \mathbf{c}_2^*), \rho^*, \Pi^*, sig^*, \mathbf{v}^*, t_r^*, t_s^*)$ on message M^*. B opens the signature Σ^* and identify the index. The improved Forking Lemma [4] ensures that, with probability at least $1/2$, B can get 3-fork involving tuple $(M, \{CMT^{(k)}\}_{k=1}^t, \mathbf{c}_1, \mathbf{c}_2)$ running A up to $32 \cdot Q_H/(\varepsilon - 3^{-t})$ times with the same records. Namely, the first $\kappa^* - 1$ random oracle queries A returns the answer given for $Ch_1, \ldots, Ch_{\kappa^*-1}$ as the first run. From the $\kappa^* - th$ query onwards A obtains a fresh random oracle values $Ch'_{\kappa^*}, \ldots, Ch'_{Q\mathcal{H}_2}$ at each new run. A simple computation shows that: $Pr[\exists j \in \{1, \ldots, t\}] : \{Ch_i^{(1)}, Ch_i^{(2)}, Ch_i^{(3)}\} = \{1, 2, 3\}$ with probability $1 - (7/9)^t$. Under the condition of the existence of such index i, one parses the 3 forgeries corresponding to the fork branches to obtain $(RSP_i^{(1)}, RSP_i^{(2)}, RSP_i^{(3)})$.

We can use the knowledge extractor of the underlying argument system to get vectors $(\mathbf{x}, \mathsf{bin}(\mathbf{z}), \mathbf{r}, \mathbf{s}, \mathbf{e}_1, \mathbf{e}_2)$ which satisfy the conditions of underlying in-

teractive protocol. Then B can get a vector which is the secret signing key of the forgery. Furthermore this vector is an answer for the SIS problem.

This confirms the traceability of the new scheme.

5.4 Non-frameability

Theorem 3: *If **SIS** problem is hard, then the new scheme is non-frameable in the random oracle model.*

We take an adversary A who can create a forgery signature that opens to a honest user i who did not generate it. Then we make a **PPT** algorithm B that figure out a solution for *SIS* problem. B knows authority keys and B creates the group public key **gpk** truly. Then B replies to the queries made by A. A can work as a corrupted group manager and registers a new user i to the group. With user i if A asks to make a signature on a message M, then B creates and returns the signature Σ=($\mathbf{ovk}, (\mathbf{c}_1, \mathbf{c}_2), \rho, \Pi, sig, \mathbf{v}, t_r, t_s$). Finally, A outputs (M^*, Σ^*=($\mathbf{ovk}^*, (\mathbf{c}_1^*, \mathbf{c}_2^*), \rho^*, \Pi^*, sig^*, \mathbf{v}^*, t_r^*, t_s^*$), which opens to i^* who did not create Σ^*. B has a short vector $\mathbf{z}_{i^*} = \mathbf{F} \cdot \mathbf{x}_{i^*} \mod q$. B should have another short vector $\mathbf{z}_{i\prime} = \mathbf{F} \cdot \mathbf{x}_{i\prime} \mod q$ to solve SIS instance. To obtain such a vector, B replays A sufficient times and applies Improved Forking Lemma [4]. Then, B can get a short vector $\bar{\mathbf{x}}$, where $\mathbf{z}_{i^*} = \mathbf{F} \cdot \bar{\mathbf{x}} \mod q$. Moreover, $\bar{\mathbf{x}} \neq \mathbf{x}_{i^*}$ with overwhelming probability according to Stern-like proof of knowledge. A nonzero vector $\mathbf{h} = \mathbf{x}_{i^*} - \bar{\mathbf{x}}$ is an answer for SIS problem.

This confirms the non-frameability of the new scheme.

6 Conclusion

In this paper, we presented a VLR group signature scheme with time-bound keys that can be applied in the hotel guest-management system. The proposed scheme used an existing scheme and discussed the advantages and security when applying the time-bound keys. As a result, this paper provided a lattice-based VLR group signature scheme with time-bound keys. The proposed scheme can be apply for systems like staff management, subscribers of a telephone company, and visitors accessing facilities. However, this paper did not present an implementation of the proposed system.

Acknowledgments. This work is supported in part by JSPS Grant-in-Aids for Scientic Research (A) JP16H01705 and for Scientic Research (B) JP17H01695.

References

1. Agrawal, S., Boyen, X., Vaikuntanathan, V., Voulgaris, P., Wee, H.: Functional encryption for threshold functions (or fuzzy ibe) from lattices. In: PKC 2012, LNCS. vol. 7293, pp. 280–297. Springer (2012)

2. Ajtai, M.: Generating hard instances of lattice problems. In: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. pp. 99–108. ACM (1996)
3. Bellare, M., Micciancio, D., Warinschi, B.: Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In: EUROCRYPT 2003, LNCS. vol. 2656, pp. 614–629. Springer (2003)
4. Brickell, E., Pointcheval, D., Vaudenay, S., Yung, M.: Design validations for discrete logarithm based signature schemes. In: PKC 2000, LNCS. vol. 1751, pp. 276–292. Springer (2000)
5. Camenisch, J., Neven, G., Rückert, M.: Fully anonymous attribute tokens from lattices. In: SCN 2012, LNCS. vol. 12, pp. 57–75. Springer (2012)
6. Chaum, D., Van Heyst, E.: Group signatures. In: EUROCRYPT 1991, LNCS. vol. 547, pp. 257–265. Springer (1991)
7. Chow, S.S., Wong, D.S.: Anonymous identification and designated-verifiers signatures from insecure batch verification. In: EuroPKI 2007, LNCS. vol. 4582, pp. 203–219. Springer (2007)
8. Chu, C.K., Liu, J.K., Huang, X., Zhou, J.: Verifier-local revocation group signatures with time-bound keys. In: Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security. pp. 26–27. ACM (2012)
9. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: ACM 2008. pp. 197–206. ACM (2008)
10. Gordon, S.D., Katz, J., Vaikuntanathan, V.: A group signature scheme from lattice assumptions. In: ASIACRYPT 2010, LNCS. vol. 6477, pp. 395–412. Springer (2010)
11. Kawachi, A., Tanaka, K., Xagawa, K.: Concurrently secure identification schemes based on the worst-case hardness of lattice problems. In: ASIACRYPT 2008, LNCS. vol. 5350, pp. 372–389. Springer (2008)
12. Laguillaumie, F., Langlois, A., Libert, B., Stehlé, D.: Lattice-based group signatures with logarithmic signature size. In: ASIACRYPT 2013, LNCS. vol. 8270, pp. 41–61. Springer (2013)
13. Langlois, A., Ling, S., Nguyen, K., Wang, H.: Lattice-based group signature scheme with verifier-local revocation. In: PKC 2014, LNCS. vol. 8383, pp. 345–361. Springer (2014)
14. Libert, B., Ling, S., Mouhartem, F., Nguyen, K., Wang, H.: Signature schemes with efficient protocols and dynamic group signatures from lattice assumptions. In: ASIACRYPT 2016, LNCS. vol. 10032, pp. 373–403. Springer (2016)
15. Ling, S., Nguyen, K., Wang, H., Xu, Y.: Lattice-based group signatures: Achieving full dynamicity with ease. In: ACNS 2017, LNCS. vol. 10355, pp. 293–312. Springer International Publishing, Cham (2017)
16. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: EUROCRYPT 2012. vol. 7237, pp. 700–718. Springer (2012)
17. Naor, D., Shenhav, A., Wool, A.: One-time signatures revisited: Have they become practical? IACR Cryptology ePrint Archive 2005, 442 (2005)
18. Peikert, C.: A decade of lattice cryptography. Foundations and Trends in Theoretical Computer Science 10(4), 283–424 (2016), https://doi.org/10.1561/0400000074
19. Perera, M.N.S., Koshiba, T.: Zero-knowledge proof for lattice-based group signature schemes with verifier-local revocation. In: NBiS 2018, LNDECT. vol. 22, pp. 772–782 (2019)
20. Perera, M.N.S., Koshiba, T.: Achieving almost-full security for lattice-based fully dynamic group signatures with verifier-local revocation. In: ISPEC 201, LNCS. vol. 11125, pp. 229–247. Springer (2018)

21. Perera, M.N.S., Koshiba, T.: Fully dynamic group signature scheme with member registration and verifier-local revocation. In: ICMC 2018, PROMS. vol. 253, pp. 399–415. Springer (2018)
22. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC 2005. pp. 84–93. ACM Press (2005)