VLR Group Signatures with Indisputable Exculpability and Efficient Revocation

Liqun Chen
HP Labs
Bristol, UK
Email: liqun.chen@hp.com

Jiangtao Li
Intel Labs
Hillsboro, USA
Email: jiangtao.li@intel.com

Abstract—Group signatures have been studied for nearly two decades and have wide applications, such as anonymous authentication and trusted computing. In 2004 Boneh and Shacham first formalized the concept of a group signature scheme with Verifier-Local Revocation (VLR) in which the revocation list is distributed only to the verifiers. This is an interesting revocation model as the signers are not involved in the revocation check and are transparent to the revocation process. Most of the existing VLR group signature schemes including the Boneh and Shacham’s original one do not satisfy exculpability without fully trusted issuer or largely increasing cost. In this paper, we propose a modification to the Boneh and Shacham VLR group signature definition by adding a dispute process to achieve exculpability. We propose a concrete VLR group signature scheme under this new definition. Our scheme is more efficient than previous VLR group signatures schemes, and only takes one exponentiation per revocation check whereas the existing VLR group signature schemes require at least one bilinear map operation per revocation check. Security of our scheme is based on the strong Diffie-Hellman assumption and the decisional Diffie-Hellman assumption in the random oracle model.

Index Terms—group signatures; trusted computing; privacy; anonymity; verifier local revocation

I. INTRODUCTION

The concept of group signatures was first introduced by Chaum and van Heyst [15] in 1991. In a group signature scheme, all members in one group share a group public key, whilst each member has his unique private key. A signature created by a group member is anonymous to verifiers but traceable to the associated group manager. Since the first introduction, this primitive has drawn lots of attention in both academic and industry. Many group signature schemes have been proposed, e.g., in [1], [7], [8], [18]. The Trusted Computing group (TCG), a global industrial standard body [23], has made use of a special case of a group signature scheme, namely Direct Anonymous Attestation (DAA) [10], to achieve computer platform authentication preserving user privacy. Recently, ISO/IEC has started to develop a new international standard on anonymous digital signatures including group signature schemes.

Like every cryptographic primitive, a robust group signature scheme should support group membership revocation. There are a number of different revocation approaches in the literature, such as updating a group public key in a reasonable internal and redistributing membership credentials only to legitimate members [2] or broadcasting a piece of revocation information to all signers and verifiers and only unrevoked signers are able to update their membership [12]. Boneh and Shacham [8] formalized the concept of a group signature scheme with Verifier-Local Revocation (VLR), in which the revocation list is distributed only to verifiers and the revocation check process is transparent to signers. This type of group signature schemes is very appealing to those applications where signers are computationally weak devices such as mobile devices, smart phones, trusted platform modules (TPM), and smart cards.

In the same paper, Boneh and Shacham proposed a concrete VLR group signature scheme. In order to achieve efficiency, their scheme requires a fully trusted key issuer to create each group member’s private key. If the trustworthy of the key issuer is arguable, then their scheme does not satisfy exculpability. The property of exculpability was first introduced by Ateniese and Tsudik in [3], with which no member of the group and not even the group manager (the key issuer) can produce signatures on behalf of other group members. It has been well-known in the literature that a group signature scheme without exculpability are not suitable for those applications in which the property of signer non-repudiation is required, since a malicious signer can simply argue that she is not the sole entity who knows the private signing key.

Recently, researchers have provided a number of different VLR group signature schemes, for instance [21], [25], [24]. The scheme of [21] extends the scheme of [8] by adding backward unlinkability without changing the trust requirement on the key issuer. The scheme of [25] satisfies the property of exculpability with the cost that checking each revoked signer requires at least one bilinear map operation. The scheme of [24] satisfies the property of exculpability very efficiently, but their scheme requires each signer only creates one signature in a certain interval. If a signer creates multiple signatures, these signatures are linkable, which to our understanding is an unacceptable requirement for most of group signature applications.

Note that it is not easy to have reasonably good performance and exculpability in a VLR group signature scheme. The target of our work presented in this paper is to find out a balance between exculpability and revocation performance in a VLR group signature scheme. As a result, we propose a new way...
to achieve exculpability. We modify the Boneh and Shacham VLR group signature definition by adding a dispute process as to answer whether a traced user is a true signer or not. In such a dispute process, an honest user can prove his identity is not bound with an inappropriate signature. Since lack of a good name, we call this type of exculpability property *indisputable exculpability*. The advantage is that it is not relied on the existence of a fully trusted private key issuer, and a true signer cannot deny that a given group signature was created under his private key and membership credential. We present a security model for VLR group signature schemes with the indisputable exculpability.

We also present a concrete scheme of VLR group signatures with indisputable exculpability along with brief security analysis (full proof will be included in the full paper) and performance comparison with the existing VLR group signature schemes. The comparison shows that our concrete scheme is more efficient than those VLR group signature schemes with or without exculpability. In particular, our scheme only takes one exponentiation per revocation check whereas the existing VLR group signatures require at least one pairing operation per revocation check. A pairing operation is about ten times or more expensive than an exponentiation operation. The revocation check in our VLR group signature scheme is fixed-base exponentiation which can be further optimized using the fast exponentiation technique [9]. Thus the revocation check in our scheme is about two orders of magnitude more efficient than the existing VLR group signature schemes. Security of our scheme is based on the strong Diffie-Hellman assumption and the decisional Diffie-Hellman assumption in the random oracle model.

Rest of this paper is organized as follows. We first give a formal specification of VLR group signatures with indisputable exculpability and present the security requirements in Section II. We review the definition of pairing and related security assumptions in Section III. Next we describe the construction of our group signature scheme in Section IV. We present an alternative scheme of our construction in Section V. We compare our construction with the existing VLR group signature schemes in Section VI. We conclude our paper in Section VII.

II. DEFINITION AND SECURITY REQUIREMENTS

Throughout the paper, we use the following notations. Let $S$ be a finite set, and $x \leftarrow S$ denote that $x$ is chosen uniformly at random from $S$. Let $b \leftarrow A(a)$ denote an algorithm $A$ that is given input $a$ and outputs $b$. Let $(c, d) \leftarrow P_{A,B}(a, b)$ denote an interactive protocol between $A$ and $B$, where $A$ inputs $a$ and $B$ inputs $b$, in the end $A$ obtains $c$ and $B$ obtains $d$.

A. Definition of VLR Group Signatures

The definition of a VLR group signature scheme is presented as follows. It is different from the definition in [8] as our group signature definition supports *indisputable exculpability*, in which when dispute happens, an honest user can prove his identity is not bound to a given group signature. A group signature scheme involves three types of entities: an issuer $I$, a set of members $M$, and verifiers $V_j$. There are the following algorithms $\text{Setup}$, $\text{Sign}$, $\text{Verify}$, $\text{Open}$, $\text{DProve}$, $\text{DVerify}$, and an interactive protocol $\text{Join}$. The algorithms $\text{DProve}$ and $\text{DVerify}$ are used for indisputable exculpability.

**Setup** This setup algorithm for the issuer $I$ takes a security parameter $1^k$ as input and outputs a group public key $\text{gpk}$ and the issuer’s private key $\text{isk}$.

$$(\text{gpk}, \text{isk}) \leftarrow \text{Setup}(1^k)$$

**Join** This join protocol is an interactive protocol between the issuer $I$ and a member $M_i$. $M_i$ has $\text{gpk}$ as input. $I$ has $\text{gpk}$, isk, and a tracing database $\text{DB}$. In the end, $M_i$ outputs an identity $\text{id}_i$, a secret key $\text{sk}_i$, and a membership credential $\text{cre}_i$ which includes a tracing key $\text{tk}_i$. $I$ outputs and appends ($\text{id}_i, \text{tk}_i$) to its tracing database $\text{DB}$. We use $\mathcal{D}B'$ to denote the updated tracing database.

$$\langle \text{DB}', (\text{id}_i, \text{sk}_i, \text{cre}_i) \rangle \leftarrow \text{Join}_{I, M}(\text{gpk}, \text{isk}, \text{DB}, \text{gpk})$$

When a member is no longer legitimate, his tracing key $\text{tk}_i$ is put into a revocation list $\text{RL}$, which is available to all verifiers.

**Sign** On input of $\text{gpk}$, $\text{sk}_i$, $\text{cre}_i$, and a message $m$, the probabilistic signing algorithm outputs a signature $\sigma$. We often call $(\text{sk}_i, \text{cre}_i)$ the signing key.

$$\sigma \leftarrow \text{Sign}(\text{gpk}, \text{sk}_i, \text{cre}_i, m)$$

**Verify** On input of $\text{gpk}$, a message $m$, a signature $\sigma$, a list of revoked tracing keys $\text{RL}$, this verification algorithm outputs valid or invalid.

$$\text{valid/invalid} \leftarrow \text{Verify}(\text{gpk}, m, \sigma, \text{RL})$$

**Open** On input of $\text{gpk}$, a message $m$, a signature $\sigma$, and the tracing database $\text{DB}$, the deterministic open algorithm outputs $\perp$ if the signature is not valid, $\perp'$ if the signature cannot be traced, or $(\text{id}_i, \text{tk}_i)$, the identity and tracing key of the signer.

$$\perp / \perp'/ (\text{id}_i, \text{tk}_i) \leftarrow \text{Open}(\text{gpk}, m, \sigma, \text{DB})$$

**DProve** On input of $\text{gpk}$, a signing key $(\text{sk}_i, \text{cre}_i)$, an identity and tracing key pair $(\text{id}_i, \text{tk}_i)$, and a message $m$, a signature $\sigma$, this dispute proof algorithm can outputs either a proof that $\sigma$ was not created using $(\text{sk}_i, \text{cre}_i)$ associated with $(\text{id}_i, \text{tk}_i)$ or $\perp$.

$$\text{proof} / \perp \leftarrow \text{DProve}(\text{gpk}, \text{sk}_i, \text{cre}_i, \text{id}_i, \text{tk}_i, m, \sigma)$$

**DVerify** On input of $\text{gpk}$, $\text{id}_i$, $\text{tk}_i$, a message $m$, a signature $\sigma$, and a dispute proof, this dispute verification algorithm outputs valid or invalid.

$$\text{valid/invalid} \leftarrow \text{DVerify}(\text{gpk}, \text{id}_i, \text{tk}_i, m, \sigma, \text{proof})$$
Note that the result valid indicates that the verifier is convinced that the owner of \((\text{id}_i, \text{tk}_i)\) is not the real signer of \(\sigma\).

One implication of verifier-local revocation is that the signature is selfless-anonymous, i.e., a member can tell whether he generated a particular signature \(\sigma\), but if he did not he learns nothing about the signer of \(\sigma\).

### B. Security Requirements

The standard definition of group signatures for dynamic groups has been proposed in [5]. The security requirements defined here are slightly different from the ones in [5] as traditional group signature schemes do not have verifier-local revocation.

A secure VLR group signature scheme needs to be correct, selfless-anonymous, traceable, and exculpable. Roughly speaking, selfless-anonymity guarantees that no one except the issuer and the signer himself can identify the creator of a group signature. Traceability guarantees that only the group opening manager can trace a group signature to its true signer. Exculpability guarantees that an honest member cannot be framed, i.e., if an attacker tries to create a group signature linked to the honest member, the honest member can dispute with a convincing proof.

1) Correctness: The correctness requirement states that every signature created by a member can be verified as valid, except if the member is revoked. It can be formally stated as

\[
\sigma = \text{Sign}(gpk, sk_i, cre_i, m) \quad \text{Valid} = \text{Verify}(gpk, m, \sigma, RL) \iff tk_i \notin RL
\]

In addition, a member can successfully dispute a signature if the signature was created using a different signing key. So, if \(i \neq j\),

\[
\sigma^* = \text{Sign}(gpk, sk_j, cre_j, m) \\
\text{proof} = \text{DProve}(gpk, sk_i, cre_i, \text{id}_i, tk_i, \sigma^*, m) \\
\text{Valid} = \text{DVerify}(gpk, \text{id}_i, tk_i, \sigma^*, \text{proof}, m)
\]

2) Selfless-Anonymity: A VLR group signature scheme is selfless-anonymous if no polynomial-time adversary can win the anonymity game. In the anonymity game, the goal of the adversary is to determine which one out of the two members generated a signature. As mentioned earlier, given a signature and a signing key, the adversary can determine whether the signature was generated using the signing key. Thus the adversary should not be given access to the signing key of either member. The anonymity game between a challenger \(C\) and an adversary \(A\) is defined as follows.

1) Setup. \(A\) runs \((gpk, i\text{sk}) \leftarrow \text{Setup}(1^k)\) and sends \(gpk\) and \(i\text{sk}\) to \(C\).

2) Queries. \(A\) can make the following queries to \(C\).
   a) Join. \(A\) requests for creating a new member \(M\). \(C\) runs the join protocol locally and generates a signing key \((sk, cre)\).
   b) Sign. \(A\) requests a signature on a message \(m\) for a member \(M\). \(C\) finds \(M\)'s signing key, computes \(\sigma \leftarrow \text{Sign}(gpk, sk, cre, m)\), and returns \(\sigma\) to \(A\).
   c) Corrupt. \(A\) requests the signing key of a member \(M\). \(C\) responds with \((sk, cre)\) of \(M\) to \(A\).
   d) Open. \(A\) requests for opening a signature \(\sigma\) on a message \(m\). \(C\) runs the open algorithm and returns the corresponding tracing key to \(A\).

3) Challenge. \(A\) outputs a message \(m\) and two members \(M_0\) and \(M_1\). \(A\) has not made a corrupt query on either member and has not made an open query on any signatures created by either \(M_0\) or \(M_1\). \(C\) chooses a random \(b \leftarrow \{0, 1\}\), computes \(\sigma \leftarrow \text{Sign}(gpk, sk_b, cre_b, m)\) where \((sk_b, cre_b)\) is the signing key for \(M_b\), and sends \(\sigma\) to \(A\).

After the challenge phase, \(A\) can make additional queries to \(C\).

4) Restricted Queries. Throughout the game, \(A\) cannot make the corrupt query on either \(M_0\) or \(M_1\) and also cannot make the open query on \(\sigma\).

5) Output. Finally, \(A\) outputs a bit \(b'\). The adversary wins if \(b' = b\).

**Definition 1:** Let \(A\) be the adversary that plays the game above. Let \(\text{Adv}[A_{\text{GS}}]\) = \(|\text{Pr}[b = b'] - 1/2|\) denote the advantage of \(A\) winning the anonymity game. The probability is taken over the coin tosses of \(A\), of the randomized setup, join, and sign algorithms, and over the choice of \(b\). We say that a VLR group signature scheme is selfless-anonymous if for any probabilistic polynomial-time adversary, \(\text{Adv}[A_{\text{GS}}]\) is negligible.

3) Traceability: A VLR group signature scheme is traceable if no adversary can win the following traceability game. In the traceability game, the adversary’s goal is to forge a valid signature that cannot be opened properly. The traceability game between a challenger \(C\) and an adversary \(A\) is defined as follows.

1) Setup. \(C\) runs \((gpk, i\text{sk}) \leftarrow \text{Setup}(1^k)\) and sends \(gpk\) to \(A\). \(C\) sets an empty revocation list \(RL\).

2) Queries. \(A\) can make the following queries to \(C\).
   a) Join. \(A\) requests for creating a new member \(M\) using one of the following two types: (1) \(C\) runs the join protocol as the issuer with \(A\) as the member. In the end, \(C\) outputs \((\text{id}, \text{tk})\) while \(A\) outputs \((\text{id}, sk, cre)\). \(C\) appends \(\text{tk}\) to the revocation list \(RL\). (2) \(C\) runs the join protocol locally and generates a signing key \((sk, cre)\).
   b) Corrupt. \(A\) requests the signing key of a member \(M\). \(C\) responds with \((sk, cre)\) of \(M\) to \(A\). \(C\) also appends \(\text{tk}\) in \(cre\) to \(RL\).
   c) Sign. The same as in the anonymity game.
   d) Open. The same as in the anonymity game.

3) Response. Finally, \(A\) outputs a message \(m\) and a signature \(\sigma\).

Assume that \(A\) did not obtain \(\sigma\) by making a sign query on \(m\). \(A\) wins if \(\text{Verify}(gpk, \sigma, m, RL) = \text{valid}\).
Definition 2: Let $\mathcal{A}$ denote an adversary that plays the traceability game above. We use $\text{Adv}[A_{\text{GS}}] = \Pr[\mathcal{A} \text{ wins}]$ to denote the advantage that $\mathcal{A}$ wins the traceability game. A VLR group signature scheme is traceable if for any probabilistic polynomial-time adversary, $\text{Adv}[A_{\text{GS}}]$ is negligible.

4) Exculpability: A VLR group signature scheme is exculpable if no adversary can win the following game. In the exculpability game, the adversary’s goal is to frame a member, i.e., create a signature that is linked to an honest member such that the member cannot dispute. The exculpability game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ is defined as follows.

1) Setup. $\mathcal{A}$ runs $(gpk, isk) \leftarrow \text{Setup}(1^k)$ and sends $gpk$ to $\mathcal{C}$. $\mathcal{C}$ sets an empty revocation list $RL$.

2) Queries. $\mathcal{A}$ can make the following queries to $\mathcal{C}$.

a) Join. $\mathcal{A}$ requests for creating a new member $M$. $\mathcal{C}$ runs the join protocol as the member with $M$ as the issuer. In the end, $\mathcal{C}$ outputs the signing key of member identity $\text{id}$, $sk$, $cre$. $\mathcal{C}$ appends $(\text{id}, tk)$ to $DB$.

b) Corrupt. $\mathcal{A}$ requests the signing key of a member $M$. $\mathcal{C}$ responds with $(sk, cre)$ to $\mathcal{M}$ to $\mathcal{A}$, and appends $tk$ in $cre$ to $RL$.

c) Sign. The same as in the anonymity game.

d) Open. The same as in the anonymity game.

3) Response. Finally, $\mathcal{A}$ outputs a message $m$, a signature $\sigma$, an identity and a tracing key pair $(\text{id}, tk)$.

Assume that $\mathcal{A}$ did neither corrupt the signer of $\sigma$ nor obtain $\sigma$ from making a sign query on $m$. Let $(sk, cre)$ be the signing key corresponding to the member identity $\text{id}$. $\mathcal{A}$ wins the game if $\text{Verify}(gpk, \sigma, m, RL) = \text{valid}$. $\text{Open}(gpk, m, \sigma, DB) = (\text{id}, tk)$, and either $\text{DProve}(gpk, sk, cre, \text{id}, tk, \sigma, m) = \perp$ or $\text{DProve}(gpk, sk, cre, \text{id}, tk, \sigma, m) = \text{proof}$ and $\text{DVerify}(gpk, \text{id}, sk, \sigma, \text{proof}, m) = \text{invalid}$.

Definition 3: Let $\mathcal{A}$ denote an adversary that plays the exculpability game above. We use $\text{Adv}[A_{\text{GS}}] = \Pr[\mathcal{A} \text{ wins}]$ to denote the advantage that $\mathcal{A}$ wins the exculpability game. A VLR group signature scheme is exculpable if for any probabilistic polynomial-time adversary, $\text{Adv}[A_{\text{GS}}]$ is negligible.

III. PAIRINGS AND COMPLEXITY ASSUMPTIONS

In this section, we first review the concept of pairings and then discuss some complexity assumptions related to our scheme.

A. Bilinear Maps

We follow the notation of Boneh, Boyen, and Shacham [7] to review the basic concept of pairings. Let $G_1$ and $G_2$ to two multiplicative cyclic groups of prime order $p$. Let $g_1$ be a generator of $G_1$ and $g_2$ a generator of $G_2$. We say $t : G_1 \times G_2 \rightarrow G_T$ is an admissible bilinear map, if it satisfies the following properties:

1) Bilinear. For all $u \in G_1, v \in G_2$, and for all $a, b \in \mathbb{Z}$, $t(u^a, v^b) = t(u, v)^{ab}$.

2) Non-degenerate. $t(g_1, g_2) \neq 1$ and is a generator of $G_T$.

3) Computable. There exists an efficient algorithm for computing $t(u, v)$ for any $u \in G_1, v \in G_2$.

We call the two groups $(G_1, G_2)$ in the above a bilinear group pair. In the rest of this paper, we consider bilinear maps $t : G_1 \times G_2 \rightarrow G_T$ where $G_1$, $G_2$, and $G_T$ are multiplicative groups of prime order $p$.

B. Strong Diffie-Hellman Assumption

The security of our group signature scheme is based on the hardness of the $q$-SDH problem introduced by Boneh and Boyen [6]. Let $G_1$ and $G_2$ be two cyclic groups of prime order $p$, respectively, generated by $g_1$ and $g_2$. The $q$-Strong Diffie-Hellman ($q$-SDH) problem in $(G_1, G_2)$ is defined as follows: Given a $(q+3)$-tuple of elements $(g_1, g_1^x, \ldots, g_1^y, g_2, g_2^x)$ as input, output a pair $(g_1^{1/(\gamma+x)}, x)$ where $x \in \mathbb{Z}_p^*$. An algorithm $A$ has the advantage $\epsilon$ in solving $q$-SDH problem in $(G_1, G_2)$ if

\[ \Pr[A(g_1, g_1^y, \ldots, g_1^{1/(\gamma+x)}, g_2, g_2^x) = (g_1^{1/(\gamma+x)}, x)] \geq \epsilon \]

where the probability is over the random choice of $\gamma$ and the random bits of $A$.

Definition 4: We say that the $(q, t, \epsilon)$-SDH assumption holds in $(G_1, G_2)$ if no $t$-time algorithm has the advantage at least $\epsilon$ in solving the $q$-SDH problem.

The $q$-SDH assumption was used by Boneh and Boyen [6] to construct a short signature scheme without random oracles and was shown in the same paper that the $q$-SDH assumption holds in the generic group in the sense of Shoup [22]. The $q$-SDH assumption was later used in [7] for constructing a short group signature scheme. The security of the SDH problem was studied by Cheon [17].

C. Decisional Diffie-Hellman Assumption

Let $G$, generated by $g$, be a cyclic group of prime order $p$. The Decisional Diffie-Hellman (DDH) problem in $G$ is defined as follows: Given a tuple of elements $(g, g^a, g^b, g^c)$ as input, output 1 if $c = ab$ and 0 otherwise. An algorithm $A$ has the advantage $\epsilon$ in solving the DDH problem in $G$ if

\[ |\Pr[g \leftarrow G, a, b \leftarrow \mathbb{Z}_p : A(g, g^a, g^b, g^{ab}) = 1] - \Pr[g \leftarrow G, a, b, c \leftarrow \mathbb{Z}_p : A(g, g^a, g^b, g^c) = 1]| \geq \epsilon \]

where the probability is over the uniform random choice of the parameters to $A$ and over the random bits of $A$.

Definition 5: We say that the $(t, \epsilon)$-DDH assumption holds in $G$ if no $t$-time algorithm has the advantage at least $\epsilon$ in solving the DDH problem in $G$.

Let $(G_1, G_2)$ be a bilinear group pair. Our proposed group signature scheme requires the DDH problem for $G_1$ to be hard. The DDH assumption on $G_1$ is also known as the External Diffie-Hellman (XDH) assumption.

IV. PROPOSED GROUP SIGNATURE SCHEME

In this section, we first present our construction of a VLR group signature scheme from bilinear maps, which satisfies the property of indisputable exculpability. Our construction builds
on top of the recent pairing-based DAA schemes [16], [11], and Furukawa and Imai group signature scheme [18]. We then discuss the efficiency of our scheme and sketch the security proof.

A. Our Group Signature Scheme

We first give some intuitions about our construction. In our group signature scheme, each member chooses a random sk := f and then obtains cre = (A, x) from the issuer such that \( i(A, wg^f) = i(g_1 h_1^f, g_2) \), where x is the tracing key. The issuer knows every member’s x value but does not know the associated secret key f. To create a group signature, the member first chooses a random B and computes J := Bi and K := Bx. The member then proves in zero-knowledge

\[
PK\{(f, A, x) : i(A, g_2^x w) = i(g_1 h_1^f, g_2) \land J = B^i \land K = B^x \}.
\]

The (B, K) pair serves not only the purpose of revocation check but also the purpose of signature tracing, as the issuer has a database of all x values and can find out which x was used in generating the signature. The J value is used for the indisputable exculpability property as only the member knows f. To prove \( i(A, wg^x) = i(g_1 h_1^f, g_2) \), the signer first computes \( T = A \cdot h_2^x \) where a is randomly chosen, and then proves the following equation as

\[
\hat{i}(T, g_2)^{-x} \cdot \hat{i}(h_1, g_2)^f \cdot \hat{i}(h_2, g_2)^{ax} \cdot \hat{i}(h_2, w)^a = \hat{i}(T, w)^a/\hat{i}(g_1, g_2) - \hat{i}(T, g_2)^{-1}\]

The VLR group signature scheme has the following algorithms Setup, Sign, Verify, Open, DProve, DVerify, and one interactive protocol Join which are defined as follows. Note that in the following description, we omit the subscripts i and j of \( \mathcal{M}_i \) and \( \mathcal{V}_j \), where it does not cause confusion.

Setup: The setup algorithm takes the following steps:

1) On input of \( f^k \), it chooses a bilinear group pair \((G_1, G_2)\) of prime order p and a pairing function \( \hat{i} : G_1 \times G_2 \rightarrow G_T \). Let \( g_1 \) and \( g_2 \) be the generators of \( G_1 \) and \( G_2 \), respectively.
2) It picks a collision resistant hash function \( H : \{0,1\}^* \rightarrow \mathbb{Z}_p^* \).
3) It chooses \( h_1 \leftarrow G_1, h_2 \leftarrow G_1, \) and \( \gamma \leftarrow \mathbb{Z}_p^* \).
4) It computes \( w := g_2^\gamma \).
5) It computes \( T_1 := \hat{i}(g_1, g_2), T_2 := \hat{i}(h_1, g_2), T_3 := \hat{i}(h_2, g_2) \), and \( T_4 := \hat{i}(h_2, w) \).
6) It outputs the public group key and private key \((gpk, sk) := ((G_1, G_2, G_T, \hat{i}, p, g_1, g_2, h_1, h_2, w, H, T_1, T_2, T_3, T_4), \gamma)\).

Note that \( T_1, T_2, T_3, \) and \( T_4 \) are optional in \( gpk \), as they can be computed from \( g_1, g_2, h_1, h_2, w, \).

Join: The join protocol is performed by a member \( \mathcal{M} \) and the issuer \( \mathcal{I} \). \( \mathcal{M} \) Takes \( gpk \) as input and \( \mathcal{I} \) has \( gpk, sk \), and \( DB \) as input. The protocol has the following steps:

1) \( \mathcal{I} \) sends a nonce \( n_1 \in \{0,1\}^k \) as a challenge to \( \mathcal{M} \).
2) \( \mathcal{M} \) chooses at random \( f \leftarrow \mathbb{Z}_p \) and computes \( F := h_1^f \).
3) \( \mathcal{M} \) sets \( sk := f \) and \( id := F \).
4) \( \mathcal{M} \) chooses at random \( r_f \leftarrow \mathbb{Z}_p \) and computes \( R := h_1^{rf} \).
5) \( \mathcal{M} \) computes \( c := H(gpk || F || n_1) \) and \( sf := r_f + c \cdot f \pmod{p} \).
6) \( \mathcal{M} \) sets \( comm := (F, c, sf) \) as a commitment of its sk, and sends \( comm \) to \( \mathcal{I} \).
7) \( \mathcal{I} \) chooses \( R := h_1^{rf} \cdot F^{-c} \) and verifies that \( sf \in \mathbb{Z}_p \) and \( c = H(gpk || F || n_1) \).
8) \( \mathcal{I} \) chooses at random \( x \in \mathbb{Z}_p \) and computes \( A := (g_1 \cdot F)^{1/(x+c)} \).
9) \( \mathcal{I} \) sets \( cre := (A, x), tK := x \) and \( id := F \).
10) \( \mathcal{I} \) sends \( cre \) to \( \mathcal{M} \).
11) \( \mathcal{I} \) updates its tracing database \( DB \) by appending \((F, x)\).
12) \( \mathcal{M} \) verifies that \( i(A, wg^f) = i(g_1 h_1^f, g_2) \).
13) \( \mathcal{M} \) outputs \((id, sk, cre) := (F, f, (A, x))\).

As in many group signature schemes [11], [18], the above join protocol needs to be executed in a sequential manner, as \( comm \) is a proof of knowledge of the discrete logarithm of the value F. To support concurrent join, we could use verifiable encryption [13] of the f value or use the concurrent join technique described in [19] with some loss of efficiency.

Sign: On input of \( gpk, sk := f, cre := (A, x) \), a message \( m \in \{0,1\}^* \), this signing algorithm takes the following steps:

1) It chooses \( B \leftarrow G_1 \) and \( J := B^i, K := B^x \).
2) It chooses \( a \leftarrow \mathbb{Z}_p \), computes \( b := ax \pmod{p} \) and \( T := A^h_2 \).
3) It randomly picks \( r_f \leftarrow \mathbb{Z}_p \), \( r_x \leftarrow \mathbb{Z}_p \), \( r_a \leftarrow \mathbb{Z}_p \), \( r_b \leftarrow \mathbb{Z}_p \).
4) It computes

\[
R_1 := B^{rf}, \quad R_2 := B^{r^x}, \quad R_3 := \hat{i}(T, g_2)^{-r_x} \cdot T_2^{rf} \cdot T_3^{rf} \cdot T_4^{fa},
\]

5) It then computes

\[
c := H(gpk || B || J || K || T || R_1 || R_2 || R_3 || m).
\]

6) It computes in \( \mathbb{Z}_p \)

\[
s_f := r_f + cf, \quad s_x := r_x + cx, \quad s_a := ra + ca, \quad s_b := rb + cb.
\]

7) It outputs \( \sigma := (B, J, K, T, c, s_f, s_x, s_a, s_b) \).

Note that \( \hat{i}(A, g_2) \) in the computation of \( R_3 \) can be pre-computed and re-used.

Verify: On input of \( gpk \), a message \( m \), a signature \( \sigma = (B, J, K, T, c, s_f, s_x, s_a, s_b) \), a list of revoked secret keys \( RL \), the verifying algorithm has the following steps:

1) It verifies that \( B, J, K, T \in G_1 \) and \( s_f, s_x, s_a, s_b \in \mathbb{Z}_p \).
2) It computes \( R_1 := B^{rf} \cdot J^{-c} \).
3) It computes \( R_2 := B^{r^x} \cdot K^{-c} \).
4) It computes

\[
R_3 := \hat{i}(T, g_2)^{-s_x} \cdot T_2^{rs} \cdot T_3^{s_b} \cdot T_4^{c} \cdot \hat{i}(T, w)^- \cdot T_1^{r} \cdot T_2^{r^x} \cdot T_3^{s_a} \cdot T_4^{a}.
\]
5) It verifies that 
\[ c = H(gpk∥B∥J∥K∥T∥\tilde{R}_1∥\tilde{R}_2∥\tilde{R}_s∥m). \]
6) If any of the above steps fails, it quits and outputs invalid.
7) For each \( x' \in RL \), if \( K = B^{x'} \), it quits and outputs invalid.
8) If none of the above steps fails, it outputs valid.

**Open:** On input of gpk, a message \( m \), a signature \( \sigma = (B, J, K, T, c, s_f, s_x, s_a, s_b) \), a tracing database DB, the open algorithm has the following steps:
1) If \( \text{Verify}(\text{gpk}, m, \sigma, 0) = \text{invalid} \), it quits and outputs invalid.
2) For each identity and tracing key pair \((i, t, k) = (F, x)\) in DB, it computes \( K' = B^\sigma \). If \( K = K' \), it outputs \((i, t, k)\).
3) If no any entry in DB matches, it outputs \( \bot \).

**DProve:** On input of gpk, sk, \( \text{cre} = (A, x), i = (F, x), t = x, k = m, \) a signature \( \sigma = (B, J, K, T, c, s_f, s_x, s_a, s_b) \), the dispute algorithm has the following steps:
1) If \( J = B^f \) and \( K = B^\sigma \), it quits and outputs invalid.
2) It computes \( \text{Sign}(\text{gpk}, \text{sk}, \text{cre}, m) \) and outputs \( \sigma' = (B', J', K', T', c', s'_f, s'_x, s'_a, s'_b) \) by following the Sign algorithm except using \( B' = B \) instead of choosing a random \( B' \in \mathbb{G}_1 \).
3) It computes \( \sigma^* := PK(f): J' = B'^f ∧ F = h_1^f \).
4) It outputs \( \text{proof} = (\sigma', \sigma^*) \).
5) **DVerify:** On input of gpk, a message \( m \), a signature \( \sigma = (B, J, K, T, c, s_f, s_x, s_a, s_b) \), an identity and tracking key pair \((F, x)\) and a proof \( \text{proof} = (\sigma', \sigma^*) \) where \( \sigma' = (B', J', K', T', c', s'_f, s'_x, s'_a, s'_b) \), the dispute verification algorithm has the following steps:
1) If it first runs \( \text{Verify} \) algorithm on \( \sigma \) and \( \sigma' \), if the algorithm outputs invalid to either of \( \sigma \) and \( \sigma' \), it also outputs invalid.
2) It verifies \( \sigma^* \); if the verification fails, it outputs invalid.
3) It verifies whether \( B = B' \) and \( K = K' \); if not, it outputs invalid.
4) If \( J = J' \), it outputs invalid; otherwise outputs valid.

**B. Efficiency of Our Scheme**

The signing key of the above scheme comprises two elements in \( \mathbb{G}_1 \) and one element in \( \mathbb{G}_1 \). The signature of the above scheme takes four elements in \( \mathbb{G}_1 \), and five elements in \( \mathbb{Z}_p \). Let \( p \) be a 256-bit prime number. Using 256-bit Barreto-Naehrig curves [4], security is approximately 128-bit and is about the same as a standard 3072-bit RSA signature. Each element in \( \mathbb{G}_1 \) is 257-bit. Thus the signing key is only 769-bit and the signature is 2308-bit in the above scheme. Using 170-bit MNT curves [20] with approximate 80-bit security, the signing key is 511-bit and the signature is 1534-bit.

Let ME denote an exponentiation or a multi-exponentiation operation which has similar computational cost. The signature generation requires five MEs in \( \mathbb{G}_1 \) and one ME in \( \mathbb{G}_T \). The signature verification takes \( 2+|RL| \) MEs in \( \mathbb{G}_1 \), one ME in \( \mathbb{G}_2 \), one ME in \( \mathbb{G}_T \), and one pairing operation. The open algorithm requires one signature verification and \( |DB| \) MEs in \( \mathbb{G}_1 \).

**C. Security Proof**

Due to the space limit, we only sketch the security proof of our VLR group signature scheme as follows.

**Theorem 1:** The group signature scheme specified above is correct.

A valid private key \((f, A, x)\) satisfies the following equation \( i(A, wg_2^x) = i(g_1h_1^f, g_2) \). Let \( T = A \cdot h_2^3 \) as computed in the step (2) of the signing algorithm. It is easy to verify the following equation holds:
\[ i(T, g_2)^{-x} \cdot T_2^f \cdot T_3^{az} \cdot T_4^a = i(T, w)/T_1. \]

Observe that the steps (3)-(6) of the signing algorithm form a proof of knowledge of \((f, x, a, b)\) such that the above equation, \( J = B^f \), and \( K = B^\sigma \) all hold. The verification algorithm essentially verifies the proof of knowledge. Therefore a group signature created by a valid group member can be successfully verified. If a group signature was created by a revoked private key \((f, A, x)\), i.e., \( x \in RL \), then the signature verification will fail in step (7) of the verification algorithm.

We now show how an honest signer can dispute a signature that was not created by him. Let \((f, A, x)\) be the signing key of the signer. Let \( \sigma = (B, J, K, T, c, s_f, s_x, s_a, s_b) \) be a group signature created using a different private key \( f \). Then \( f \neq \log_B J \) as nobody besides the signer knows the value of \( f \). The honest signer can create the dispute proof by creating a new signature \( \sigma' = (B', J', K', T', c', s'_f, s'_x, s'_a, s'_b) \) such that \( B' = B \) and \( J' = B'^f \neq J \). Thus he is able to dispute successfully.

**Theorem 2:** Under the \( \mathbb{G}_1\)-DDH assumption, the group signature scheme specified above is selfless-anonymous.

If there is an adversary \( A \) that succeeds with a non-negligible probability to win the selflessness-anonymity game, then this adversary can be used by a polynomial-time algorithm \( B \) to solve the \( \mathbb{G}_1\)-DDH problem with a non-negligible probability. Suppose \( B \) has a target that given \((P, P^\alpha, P^\beta, P^\rho) \in \mathbb{G}_1^4 \) and \( \alpha, \beta, \rho \in \mathbb{Z}_p^* \), answer whether \( \rho = \alpha \beta \) or not. \( B \) manages to let two randomly selected members \( M_0 \) and \( M_1 \), which happen to be the \( A \) challenging pair, have \( f_0 = \alpha \) and \( f_1 = \beta \). In the challenge phase, if \( b = 0, B \) chooses \( B = P \) and \( J = P^\alpha \); if \( b = 1, B \) chooses \( B = P^\beta \) and \( J = P^\rho \). If \( A \)'s output \( b' \) is equal to \( b, B \) outputs \( N_0 \); otherwise outputs \( Y e s \). The reason is that if \( \rho = \alpha \beta \), then \( f_0 = f_1, A \)'s answer is purely random; otherwise, \( A \)'s answer should be equal to the value \( b \), since we assume \( A \) is able to break the selflessness-anonymity property by distinguishing which one between \( M_0 \) and \( M_1 \) is the real signer.

**Theorem 3:** Under the \( q\)-SDH assumption, the group signature scheme specified above is traceable.

If there is an adversary \( A \) that succeeds with a non-negligible probability to win the traceability game, then \( A \) can be used by a polynomial-time algorithm \( B \) to solve
the $q$-SDH problem with a non-negligible probability. To win the traceability game means that $A$ is able to create a valid signature, which is neither from a sign query nor by a corrupted member. If this happens, $B$ can use a rewinding operation to retrieve a new pair of $(A, x)$, which will allow $B$ to solve the $q$-SDH problem, by using the same approach as used in [7].

**Theorem 4:** Under the Discrete Log (DL) assumption, the group signature scheme specified above is exculpable.

If there is an adversary $A$ that succeeds with a non-negligible probability to win the exculpability game, then $A$ can be used by a polynomial-time algorithm $B$ to solve the discrete logarithm problem with a non-negligible probability. Suppose the target of $B$ is that given a pair of elements $(g, h)$ in $G_1$, output the discrete log $\log_g h$. $B$ simulates the above group signature scheme by letting $h_1 = g$ and creating a special member with his secret key $f = \log_g h$. $B$ runs the exculpability game with $A$, in which $A$ is allowed to corrupt the group manager and any other members. In the end, the adversary $A$ wins the exculpability game, that means $A$ is able to create a signature along with a pair of an identity $F = h_1^f = h$ and a tracing key $x$, such that the signature can pass the verification algorithm, and the signature is traced to the identity and tracing key pair using the open algorithm, but the dispute prove algorithm fails. The dispute prove algorithm failure means that in the signature created by $A$, the equation $J = B^f$ holds, therefore nobody even in possession of the value $f$ can show the discrete logs of $F = h_1^f$ and $J = B^f$ to the respective bases $h_1$ and $B$ are not equal. In that case, $B$ can use a rewind to retrieve the signing key $(f, A, x)$ of the special member, where $f = \log_g h$. This allows $B$ to solve the discrete log problem.

V. AN ALTERNATIVE SCHEME

The reason that our VLR group signature scheme involves a dispute proof and verification process is because the revocation token $x$ is not cryptographically bound to the signer’s identity $F$. If the open algorithm takes as input a signature $\sigma$ and a tracing database $DB$ and outputs a result $(F, x)$, this means that the signature was signed under a membership private key $f$ and membership credential $(A = ((g_1 h_1^f)^{1/(\gamma + x)}, x))$. But whether $F = h_1^f$ or not is arguable. With the $\text{DProve}$ algorithm the owner of $F$ can prove whether this equation holds, and therefore she cannot deny the responsibility for $\sigma$ if the equation is a true statement. It looks that the dispute process is an extra operation in our scheme, comparing with the existing VLR group signature schemes, but the advantage of our solution is that the revocation operation is more efficient. We argue that since the dispute process does not have to be run often, this trade-off is worthy to have.

Note that our scheme can be changed to a traditional VLR group signature scheme without the dispute process but also supporting exculpability. Let $\psi(\cdot)$ be an efficiently computable isomorphism from $G_2$ to $G_1$. In the join process the signer sends $\hat{F} = h_1^f$ instead of $F = h_1^f$ to the issuer, where $h_1 \in G_2$ and $h_1 = \psi(h_1)$. The issuer then computes $F = \psi(\hat{F}) = h_1^f$ and computes $(A, x)$ accordingly. The issuer uses $\hat{F}$ as the tracing key. In the sign algorithm, the member proves in zero-knowledge

$$PK\{(f, A, x) : \hat{t}(A, g_2^w) = \hat{t}(g_1 h_1^f, g_2) \land J = B^f\}.$$ 

In the verification algorithm, the linkage between $J$ and $\hat{F}$ can be verified by checking $\hat{t}(J, h_1) = \hat{t}(B, F)$. Observe that if a member is indeed revoked, i.e., $\hat{F} = h_1^f$ is placed in $\mathbb{R}_L$, then $\hat{t}(J, h_1) = \hat{t}(B, h_1)^f = \hat{t}(B, F)$ and his signatures will be rejected in the verification algorithm. The cost of revocation operation is the same as the scheme of [8], however the dispute process is no longer necessary because only the group member himself knows the $f$ value which is used in the revocation check, and only the group member can compute $J$ correctly.

In this alternative scheme, the value $K = B^x$ and the related operations can be omitted, thus the size of the group signature can be further reduced.

VI. COMPARISON WITH EXISTING WORK

We compare our VLR group signature schemes with some pairing based group signature schemes in Table I. In the table, we let $\text{ME}$ denote the cost of an exponentiation or a multi-exponentiation, $\text{BM}$ denote the case of a bilinear map computation, and $\mathbb{R}_L$ denote the size of the revocation list $RL$. In the following comparisons, we use the bilinear map function in 256-bit Barreto-Naehrig curves [4] which has about 128-bit security level. Each element in $G_2$ takes 257-bit to represent and each element in $G_2^{\tau}$ takes 3072-bit to represent.

We first compare the efficiency of our scheme with the traditional group signature schemes. Boneh, Boyen, and Shacham constructed the first efficient pairing-based short group signatures [7]. Furukawa and Imai further improved the BBS group signatures with exculpability and better efficiency in [18]. Our group signature scheme has shorter signature size than [7], [18]. The computational cost of our scheme is also smaller if we treat the cost of the revocation check separately. Note that the group signature schemes [7], [18] themselves do not support any revocation mechanisms. Revocation could be added but with an additional cost.

We now compare the efficiency of our scheme with the existing VLR group signature schemes. Boneh and Shacham constructed the first VLR group signature scheme [8]. Nakashoji and Funabiki improved the VLR group signature scheme with backward unlinkability [21]. Zhou and Lin proposed the first VLR group signature scheme with exculpability [25]. Compared with these scheme, our group signature scheme has the best efficiency in sign, verify, open, and revocation check. We do not include the scheme of [24] since it does not satisfy the property of unlinkability for multiple signatures signed in the same interval. Note that an exponentiation operation is much cheaper than a bilinear map operation. The revocation check in our scheme is fix-base exponentiation which can be further optimized using the fast exponentiation technique [9]. Thus the revocation check in our scheme could be about two orders of magnitude more efficient than any of the existing
VLR group signature schemes. Since the open algorithm is similar to revocation check schemes in the construction, the open algorithm in our scheme is much efficient than the existing schemes as well. This property makes our scheme much attractive in practice.

For completeness, we also list our alternative scheme specified in Section V in the table. In this alternative scheme, the cost of revocation operation is the same as in the BS VLR group signature scheme [8], however the computation cost of the signing operation is much cheaper.

VII. CONCLUSION

We presented a new model for VLR group signatures with indisputable exculpability. In this model, there is a dispute process such that an honest signer can prove that his identity is not bound with a given signature that he did not generate even if the signature is traced to him. We also described an efficient VLR group signature from bilinear maps that satisfies this indisputable exculpability property. Our VLR group signature scheme is secure under the strong Diffie-Hellman assumption and the decisional Diffie-Hellman assumption in the random oracle model. We will present the full security analysis of our VLR group signature schemes in the full paper. Future work includes prototyping of our group signature schemes and corresponding performance analysis.

REFERENCES


