Leakage-Resilient Group Signature: Definitions and Constructions

Jianye Huang, Qiong Huang*

College of Mathematics and Informatics, South China Agricultural University, Guangzhou 510642, China.

Abstract

Group signature scheme provides group members a way to sign messages without revealing their identities. Anonymity and traceability are two essential properties in a group signature system. However, these two security properties hold based on the assumption that all the signing keys are perfectly secret and leakage-free. On the other hand, on account of the physical imperfection of cryptosystems in practice, malicious attackers can learn fraction of secret state (including secret keys and intermediate randomness) of the cryptosystem via side-channel attacks, and thus breaking the security of whole system.

To address this issue, Ono et al. introduced a new security model of group signature, which captures randomness exposure attacks. They proved that their proposed construction satisfies the security requirements of group signature scheme. Nevertheless, their scheme is only provably secure against randomness exposure and supposes the secret keys remains leakage-free. In this work, we focus on the security model of leakage-resilient group signature based on bounded leakage setting and propose three new black-box constructions of leakage-resilient group signature secure under the proposed security models.

Key words: group signature, full anonymity, full traceability, black-box construction, leakage resilience

1. Introduction

Group Signature. Group signature, introduced in the seminal work of Chaum and Heyst [16], provides group members with anonymity while signing messages. That is, every group members is able to anonymously sign messages on behalf of their group. For example, Alice is an employee (group member) of a large company. She is able to submit an anonymous tip and convinces all verifiers that it is signed by one of employees in Alice’s company without revealing her identity. On the other hand, if Alice abuses her anonymity power to sign bogus messages, her identity can be traced by her employer Bob (group manager). Briefly, a group signature scheme must satisfy the following basic security requirements [3].

Correctness. Signatures honestly generated by group members can always be verified while invalid signatures always failing verification process.
Unforgeability. Only group members can sign messages on behalf of their group. In other word, non-members cannot efficiently forge valid group signatures.

Anonymity. Given any message/signature pair, the identity of the real signer cannot be revealed without group manager’s help.

Traceability. Given any valid message/signature pair, the group manager is able to trace the real signer’s identity.

Unlinkability. Given any two message/signature pairs, no probabilistic polynomial-time distinguisher can decide if the signatures are both from the same signer or not.

Exculpability. Even if all other group members and the managers collude, they cannot forge a signature for an honest group member.

Coalition Resistance. A colluding subset of group members cannot generate a valid signature that is traced to other non-participants.

A formal definition was proposed by Bellare, Micciancio and Warinschi [8]. In their work, they proposed three security properties Correctness, Full-Anonymity and Full-Traceability. They showed that a group signature fulfilling these three properties implies the satisfaction of all the security requirements described above. More details are presented in Sect. 3.3.

Side-Channel Attacks and Leakage Models. Unfortunately, when we consider physical attacks (e.g. side-channel attack) in practice, most existing group signature schemes does not satisfy the aforementioned security requirements any more. For instance, Alice’s co-worker, say Eve, measures the power consumption of her computer while Alice is sending an anonymous tip and signing it with her secret key. Then the resulting power traces may subsequently lead to the recovery of Alice’s secret key.

Over the last twenty years, researchers have discovered a wide range of side channel attacks, including running-time attacks [34], fault detection [11], differential power analysis (DPA) [33, 18], electromagnetic radiation analysis [23, 39], and etc. In 2009, Halderman et al. [25] proposed the well-known cold-boot attack on private keys stored in the memory of devices. Such memory leakage showed that obtaining secret information (even not being used) is possible. In 2017, Craig Ramsay [40] proposed an efficient tempest attack against AES encryption algorithms. Ramsay showed how to recover the encryption keys from two realistic AES-256 implementations within one meter attack distance and few minutes. Furthermore, their equipment is only pocket-size and low-cost. Nowadays, it is feasible to obtain secret keys (or other sensitive information like intermediate randomness) from the cryptosystem via such physical attacks. On the other hand, traditional cryptography does not capture the side channel attacks that focus the weakness of devices themselves and steer clear of the intractability of mathematical hard problems. Therefore, it is a great threat for many cryptographic systems and countermeasures for protecting cryptographic systems from such physical attacks are imperative.
For this purpose, in 2008, Dziembowski et al. first introduced the notion of leakage-resilient cryptography (LRC) [21]. Micali and Reyzin [37] proposed the well-known atom, *Only Computation Leaks Information*. They assumed that only the part of secret information that is being operated will be possibly leaked to attackers. It is not trivial, however, providing a heuristic solution to formalize the leakage in practice. Inspired by Akavia, Goldwasser and Vaikuntanathan’s work [1], leakage is defined as an adversarially chosen function of private state. The adversary is given an additional leakage oracle, which on input an adversarially chosen function and return the leakage (function value) of private state of cryptographic system. In terms of the restriction of adversaries’ power, there are different leakage models. For example, in *bounded leakage model* [2, 1], there must be sufficient entropy left ensuring the system is not fully ruined. That is, the leakage amount is bounded. In *noisy leakage model* [20, 21], instead of restricting the amount of leakage to a concrete bound, arbitrarily large leakage is allowed as long as the secret information remains sufficient min-entropy. It is more accurate to model leakage in the reality, and thus can capture more practice attacks comparing with bounded leakage model. In *continual leakage model* [13, 30], the secret keys should be periodically updated without modifying the corresponding public keys, which allows the adversary to obtain in total arbitrarily many bits of the cryptosystem’s private state as long as the leakage between two invocations of the secret key refreshing algorithms is bounded. Namely, the leakage rate is bounded.

1.1. Related Works

*Group Signature*. Subsequent to the introduction of Chaum and Heysts seminal work [16], many proposed schemes, including [7, 15, 10], focused on improving the performance of Chaum-Heysts seminal construction. [4, 14, 6] investigated the dynamics property to support the member revocation. [14, 6] studied the independent-generation of group member keys with member revocation to support large group. [9] considered the forward security of group signatures. Chen and Pedersen [17] proposed the first group signature scheme with dynamical group size. Camenisch and Stadler [15] presented the first group signature scheme for large groups, in which the size of group public key and signatures is independent of the group size. To support efficient member revocation, Kim et al. [32] constructed a group signature scheme from *traitor tracing schemes*. In 2003, Ateniese and de Medeiros [5] proposed another group scheme supporting trapdoor privacy. That is, no third party knows any trapdoor secret, and therefore a same cryptographic domain can be shared in different groups without security loss. It is a great advantage over other group signature schemes.

*Leakage-Resilient Signature Primitives*. There has been impressive progress in leakage-resilient cryptography. In 2009, Katz and Vaikuntanathan [31] proposed a secure black-box construction of signature scheme in the bounded leakage model. Boyle et al. [12] and Yuen et al. [45] considered the fully leakage-resilient unforgeability, which takes account of leakage on both signing keys and private randomness. Wang et al.
[43, 44] and Huang et al. [27, 28] studied the signature schemes with strong unforgeability in different leakage settings. Huang et al. [26] proposed black-box constructions of identity-based signature scheme and certificateless signature scheme. [29] proposed another efficient method to construct a leakage-resilient signature via introducing the notation of leakage-resilient dual form signature. Many other leakage-resilient models are studied in literatures, for example, [36, 22].

Leakage-Resilient Group Signature. To the best of our knowledge, however, there are seldom works on leakage-resilient group signature. Ono et al. [38] proposed a new security model of group signature for capturing full randomness exposure and proved their proposed scheme satisfies the security requirements. Nevertheless, their scheme is only provably secure against randomness exposure and the secret keys is still assumed to be leakage-free. How to construct a (fully) leakage-resilient group signature scheme remains an open problem.

1.2. Our Contributions

In this work, we propose three black-box constructions of group signature scheme that satisfies correctness, full anonymity, full traceability and leakage resilience at the same time. To be more concrete, we make the following contributions in this paper.

1. First of all, we revisit the security models of group signature and give a formal definition of leakage-resilient group signature scheme.

2. Second, we present a new construction of group signature scheme via black-box method, constructing from a leakage-resilient identity-based signature scheme and a CCA-secure leakage-resilient encryption scheme. We further prove that the leakage bound of resulting construction is the same as the lower-bound of the IBS scheme and the encryption scheme.

3. Third, we show how to construct a group signature scheme with leakage resilience from a leakage-resilient signature scheme and a CCA-secure leakage-resilient encryption scheme. The leakage bound of resulting construction is the same as the lower-bound of the underlying signature scheme and encryption scheme.

4. Forth, we further propose a black-box construction of group signature scheme from a leakage-resilient signature scheme and a CPA-secure leakage-resilient scheme. The leakage bound of resulting construction is the same as the lower-bound of the underlying signature scheme and encryption scheme.

2. Preliminaries and Definitions

2.1. Notations and Abbreviations

Definition 1 (Negligible Function [35]) We say a function \( \mu(x) : \mathbb{N} \rightarrow \mathbb{R} \) is negligible if for every positive polynomial \( \text{poly}(\cdot) \) there exists a positive integer \( K \in \mathbb{N} \) s.t. \( \forall k > K, |\mu(k)| < \frac{1}{\text{poly}(k)} \) holds.
Let \( k \in \mathbb{N} \) be the security parameter and \( 1^k \) be its unary representation. We assume that all the algorithms are given the unary coding of the security parameter \( k \) as input, which will not be explicitly involved as input of the algorithms when it is clear from the context. Let \( \text{negl}(k) \) denotes a negligible function of security parameter \( k \). Denote \( B(x_1, x_2, \cdots, x_n; r) \) by the output of evaluating PPT algorithm \( B \) on input \( x_1, x_2, \cdots, x_n \) with randomness \( r \). For any finite set \( S \), we denote \( \alpha \leftarrow S \) by the operation of uniformly sampling a random element \( \alpha \) from \( S \). Let \( \| \alpha \| \) be the size of element \( \alpha \). We write \([n] \) to denote the set of positive integers from 1 to \( n \), i.e. \( \{1, 2, \cdots, n\} \). We use the abbreviation PPT to mean probabilistic polynomial-time. The abbreviation KGC signifies key generation center.

2.2. Unbounded Simulation-Sound Non-Interactive Zero Knowledge

Definition 2 (Unbounded Simulation-Sound NIZK [41, 19]) An unbounded simulation-sound NIZK proof system \( \Pi = (l := l(k), P, V, S = (S_1, S_2)) \) for a \( \mathcal{NP} \)-language \( L \) with relation \( R \) involves four PPT algorithms \( P, V, S_1, S_2 \) that satisfy the following properties.

Completeness. It holds that \( \Pr[V(x, P(x, w, r), r)] = 1 : (x, w) \in \{(x, w)| R(x, w) = 1 \} \land r \in \{0, 1\}^l \} = 1 \).

Simulation Soundness. Let \( T \) be the transcripts generated by the simulation algorithm \( S_2 \) and \( \text{Succ} \) be the event \( (x, \pi) \not\in T \land x \not\in L \land V(x, \pi, r) = 1 \). Then we call \( \Pi \) satisfies simulation-soundness if the advance of any PPT adversary \( A \) succeeding in generating a valid proof on a false statement \( x \) is negligible, i.e. \( \Pr[\text{Succ} : (r, \rho) \leftarrow S_1(1^k), (x, \pi) \leftarrow A^{S_1(\cdot, \cdot)\cdot}(r)] \leq \text{negl}(k) \).

Unbounded Zero Knowledge. We call \( \Pi \) satisfies unbounded zero knowledge if for all PPT adversaries \( A \),

\[
\left| \Pr[\text{Expt}_0^A(k) = 1] - \Pr[\text{Expt}_1^A(k) = 1] \right| \leq \text{negl}(k)
\]

always holds, where \( \text{Expt}_0(k) \) and \( \text{Expt}_1(k) \) is defined as below.

\[
\begin{align*}
\text{Expt}_0(k) & \quad \begin{cases} 
    r \leftarrow \{0, 1\}^l \\
    S'_0(x, w) := P(x, w, r) \\
    \text{Return } A^{S_0^\prime(\cdot)\cdot}(r).
\end{cases} \\
\text{Expt}_1(k) & \quad \begin{cases} 
    (r, \rho) \leftarrow S_1(1^k) \\
    S'_1(x, w) := S_2(x, r, \rho) \\
    \text{Return } A^{S_0^\prime(\cdot)\cdot}(r).
\end{cases}
\end{align*}
\]

3. Definitions and Security Models of Signature Schemes

3.1. Digital Signature

Definition 3 (Digital Signature [24]) A digital signature scheme \( \Sigma \) consists of the following three polynomial time algorithms.

Key Generation Algorithm. On input a security parameter \( k \) represented in unary, the key generation algorithm produces a verification/signing key pair \( (vk, sk) \leftarrow \text{Kg}(1^k) \).
**Signing Algorithm.** On input a signing key $sk$ and a message $m$, the algorithm returns a valid signature $\sigma \leftarrow \text{Sig}(sk, m)$.

**Verification Algorithm.** On input a verification key $vk$, a message $m$ and a signature $\sigma$, the algorithm returns $1/0$ (acceptance/rejection) $\leftarrow \text{Ver}(vk, m, \sigma)$.

**Security Model.** The standard security notion of digital signature is *existential unforgeability under adaptive chosen-message attack* (UF-CMA, for short). Consider the following unforgeability experiment for a signature scheme $\Sigma = (Kg, \text{Sig}, \text{Ver})$, where $\mathcal{C}$ is a challenger and $\mathcal{F}$ is a PPT forger who tries to produce a valid signature on a new message.

**UF-CMA Experiment $\text{UF-CMA}_{\mathcal{F}, \Sigma}(k)$:**

1. $\mathcal{C}$ invokes $(vk, sk) \leftarrow Kg(1^k)$ and gives the verification key $vk$ to $\mathcal{F}$ while keeping the signing key $sk$ secret. $\mathcal{C}$ maintains an initially empty query list $L_s$.

2. $\mathcal{F}$ adaptively accesses to a signing oracle $O_s(\cdot)$ for polynomially many times. Given a message $m$, the signing oracle returns a valid signature $\sigma$ on $m$. Update the signing query list $L_s = L_s \cup \{m\}$.

3. Finally, $\mathcal{F}$ outputs a forgery $(\hat{m}, \hat{\sigma})$ and wins iff. (a) $\text{Ver}(\hat{m}, \hat{\sigma}) = 1$ and (b) $\hat{m} \notin L_s$.

**Definition 4 (Unforgeability under Adaptive Chosen-Message Attack, UF-CMA [24])** A signature scheme $\Sigma$ is existentially unforgeable under adaptive chosen-message attack (or is UF-CMA secure) if the advantage of any PPT forger $\mathcal{F}$ in the experiment $\text{UF-CMA}_{\mathcal{F}, \Sigma}(k)$ is negligible, i.e. for all PPT forger $\mathcal{F}$,

$$\Pr\left[\text{Ver}(vk, \hat{m}, \hat{\sigma}) = 1 \land \hat{m} \notin L_s : (\hat{m}, \hat{\sigma}) \leftarrow \mathcal{F}^{O_s(\cdot)}(vk)\right] \leq \text{negl}(k).$$

**Leakage-Resilient Unforgeability.** To model a forger $\mathcal{F}$ who is allowed to launch side-channel attacks against signature scheme $\Sigma$ and obtain a fraction of signing $sk$, $\mathcal{F}$ is given a leakage oracle $O_L(\cdot)$. On input the $i$th adversarially chosen leakage function $f_i$, the leakage oracle $O_L(f_i)$ returns $\Lambda_i := f_i(sk)$ where $\|\Lambda_i\| \leq \lambda_i$. Without loss of generality, we suppose that the $\mathcal{F}$ makes leakage queries at most $q_L$ times. We have following definition.

**Definition 5 (Leakage-Resilient Unforgeability [27, 29])** A signature scheme $\Sigma$ is $\lambda$-leakage-resilient and existentially unforgeable under adaptive chosen-message attacks (or is $\lambda$-LR-UF-CMA secure) if no PPT forger $\mathcal{F}$ has non-negligible advantage in the modified experiment, i.e. for all PPT forger $\mathcal{F}$,

$$\Pr\left[\text{Ver}(vk, \hat{m}, \hat{\sigma}) = 1 \land \hat{m} \notin L_s \land \sum_{i=1}^{q_L} \lambda_i \leq \lambda : (\hat{m}, \hat{\sigma}) \leftarrow \mathcal{F}^{O_s(\cdot), O_L(\cdot)}(vk)\right] \leq \text{negl}(k).$$

### 3.2. ID-based Signature

**Definition 6 (Identity-Based Signature, IBS [42])** An identity-based signature scheme $\text{IBS}$ consists of four PPT algorithms $\text{Setup}, \text{Extract}, \text{IB-Sig}, \text{IB-Ver}$, respectively called setup algorithm, key extraction algorithm, signing algorithm and verification algorithm.
Setup Algorithm. On input a security parameter $k$ represented in unary, KGC invokes setup algorithm $\text{Setup}$ to generate a master public/secret key pair $(mpk, msk) \leftarrow \text{Setup}(1^k)$.

Extraction Algorithm. On input a master key $msk$ and a user identity $id$, KGC invokes extraction algorithm $\text{Extract}$ to generate a signing key for user $id$, i.e. $sk_{id} \leftarrow \text{Extract}(msk, id)$.

Signing Algorithm. On input a user signing key $sk_{id}$ and a message $m$, the user invokes signing algorithm $\text{IB-Sig}$ to produce a signature $\sigma \leftarrow \text{IB-Sig}(sk_{id}, m)$.

Verification Algorithm. On input a user identity $id$, a message $m$ and a signature $\sigma$, the verifier invokes verification algorithm $\text{IB-Ver}$ to test the validity the purported signature, i.e. $1/0$ (acceptance/rejection) $\leftarrow \text{IB-Ver}(id, m, \sigma)$.

Remark 1 The master public key $mpk$ is implicitly involved in extraction algorithm, signing algorithm and verification algorithm and therefore it is omitted in this paper for simplicity.

Security Model. The standard security notion of IBS scheme is existential unforgeability under adaptive chosen-message and chosen-identity attack. Let $\text{IBS} = (\text{Setup}, \text{Extract}, \text{IB-Sig}, \text{IB-Ver})$ be an IBS scheme. Consider the following unforgeability experiment where $C$ is challenger and $\mathcal{F}$ is a PPT forger.

UF-CMIA Experiment $\text{CMIA-IDA}_{\mathcal{F}, \text{IBS}}(k)$:

1. $C$ generates master public/secret keys by invoking $(mpk, msk) \leftarrow \text{Setup}(k)$. Extraction query list $L_e$ and signing query list $L_s$ are initialized to be empty. Initially set system state $\text{State} = \{(KGC, msk)\}$ and leakage amount $L = 0$.

2. Invoke $\mathcal{F}(mpk)$ and answer queries from $\mathcal{F}$ as follows.

   Create User Oracle $\text{CO}$. Given a user identity $id$, if $(id, \ast) \in \text{State}$, none of the process takes place. Otherwise, generate user signing key by invoking $sk_{id} \leftarrow \text{Extract}(msk, id)$. Update the system state $\text{State} = \text{State} \cup \{(id, sk_{id})\}$.

   Extraction Oracle $E_O$. Given a user identity $id$, retrieve the corresponding $sk_{id}$ s.t. $(id, sk_{id}) \in \text{State}$ and return $sk_{id}$. Update the extraction query list $L_e = L_e \cup \{id\}$.

   Signing Oracle $S_O$. Given a user identity $id$ and a message $m$, retrieve the corresponding $sk_{id}$ from $\text{State}$ and return $\sigma \leftarrow \text{IB-Sig}(sk_{id}, m)$. Update signing query list $L_s = L_s \cup \{(id, m)\}$.

   Leakage Oracle $L_O$. Given a function $f$, compute $\Lambda := f(\text{State})$. If $L + \|\Lambda\| > \lambda$, abort. Otherwise, return $\Lambda$ and update the leakage amount $L = L + \|\Lambda\|$.

3. Finally, $\mathcal{F}$ outputs a forgery $(\hat{id}, \hat{m}, \hat{\sigma})$. Denote by $\text{Forge}$ the event: (a) $\text{IB-Ver}(\hat{id}, \hat{m}, \hat{\sigma}) = 1$, (b) $\hat{id} \notin L_e$ and (c) $(\hat{id}, \hat{m}) \notin L_s$.

Remark 2 Without loss of generality, suppose that each $id$ used in other oracle queries has already been created by $\text{CO}(id)$.
Definition 7 (Leakage-Resilient Identity-Based Signature, LR-IBS [26]) An IBS scheme \( IBS \) is \( \lambda \)-leakage-resilient and existentially unforgeable under adaptive chosen-message and chosen-ID attack (or is \( \lambda \)-LR-UF-CMIA secure) if no probabilistic polynomial-time forger \( F \) has a non-negligible advantage in the Experiment CMIA-IDA\(_{IBS}(k)\), i.e.

\[
\Pr \left[ \text{IB-Ver}(\hat{id}, \hat{m}, \hat{\sigma}) = 1 \middle| \hat{id} \notin L_e, (\hat{id}, \hat{m}) \notin L_s : (\hat{id}, \hat{m}, \hat{\sigma}) \leftarrow \mathcal{F}^{\mathcal{C}O(\cdot), \mathcal{L}O(\cdot), \mathcal{S}O(\cdot), \mathcal{L}O(\cdot)(vk)} \right] \leq \text{negl}(k).
\]

3.3. Group Signature

Definition 8 (Group Signature [16]) A group signature scheme \( GS \) consists of the following four polynomial-time algorithms.

\textbf{Setup}(1^k) \rightarrow (PP, msk, tsk). The setup algorithm takes as input the security parameter \( k \) (in unary representation) and outputs a group parameter \( PP \), a master key \( msk \) for member authorization and a tracing key \( tsk \) for identifying signers.

\textbf{Join}(PP, msk, id) \rightarrow sk_{id}. The join algorithm takes as input the group parameter \( PP \), the master key \( msk \) and an identity \( id \), output signing key \( sk_{id} \), which is sent to user via a secure channel.

\( GSig(PP, sk_{id}, m) \rightarrow \sigma. \) On input public parameter \( PP \), a signing key \( sk_{id} \) of user \( id \) and a message \( m \), the signing algorithm outputs a valid group signature \( \sigma \).

\( GVer(PP, m, \sigma) \rightarrow 0/1. \) On input the public parameter \( PP \), a message \( m \) and a signature \( \sigma \), if \( \sigma \) is a valid signature on \( m \) signed by one of the group members, the verification algorithm outputs 1 and 0 otherwise.

\( \text{Trace}(PP, tsk, m, \sigma) \rightarrow id/ \perp. \) The trace algorithm takes as input the public parameter \( PP \), a tracing key \( tsk \) and the message/signature pair \( (m, \sigma) \), output the real signer \( id \) who produces the signature \( \sigma \) on message \( m \). If \( id \) is not a group member, output \( \perp \).

We follow the security definition presented in Bellare, Micciancio and Warinschis work [8]. That is, a group signature must satisfy three properties: correctness, full anonymity and full traceability. Details are as follows.

Correctness. Generally, we say that a group signature scheme satisfies correctness if any honestly generated signature is always verified. I.e.

\[
\Pr \left[ \text{GVer}(PP, m, GSig(PP, sk_{id}, m)) = 1 : (m, id) \in \{0, 1\}^* \right] = 1.
\]

Security Model. Let EList, SList, CList, TList be lists of enrollment queries, signing queries, corrupt queries and tracing queries respectively and \( L \) be the leakage amount. Initially set EList = SList = CList = TList = State = \( \phi \) and \( L = 0 \). Consider the following oracles.
OEnroll The enrollment oracle takes as input an index of specific user identity $id$ and runs $sk_{id} \leftarrow \text{Join}(PP, msk, id)$. Notice that if $(id, \ast) \in \text{EList}$, none of the process takes place. Update the enrollment query list $\text{EList} = \text{EList} \cup \{(id, sk_{id})\}$ and system state $\text{State} = \text{State} \cup \{(id, sk_{id})\}$.

OSig(·) The signing oracle takes as input an index of specific user identity $id$ and a message $m$, returns the corresponding signature $\sigma$. Update the signing query list $\text{SList} = \text{SList} \cup \{(id, m, \sigma)\}$.

OCorrupt(·) The corruption oracle takes as input a specific user identity $id$ and retrieve the users signing key $sk_{id}$ s.t. $(id, sk_{id}) \in \text{EList}$. Return $sk_{id}$ and update the corruption query list $\text{CList} = \text{CList} \cup \{id\}$.

OTrace(·, ·) The tracing oracle takes as input a valid message/signature pair $(m, \sigma)$, return the identity $id$ of the real signer. Update the tracing query list $\text{TList} = \text{TList} \cup \{(m, \sigma)\}$.

OLeak(·) The leakage oracle takes as input a leakage function $f(\cdot)$ and computes $\Lambda := f(\text{State})$ where $\text{State}$ contains the master signing key and all signing keys of group members. If $L + \|\Lambda\| \leq \lambda$, return $\Lambda$ and update $L = L + \|\Lambda\|$. Otherwise, return abort symbol $\perp$ and aborts.

Remark 3 We assume all the identities submitted to signing oracle and corruption oracle have been enrolled in enrollment oracle and therefore their signing is already stored in enrollment list $\text{EList}$.

Full Anonymity. We say a group signature satisfies full anonymity if any PPT verifier learns nothing about the information of the actual signers identity from a given group signature. That is, even $t$ group members collude together, they cannot correctly identify the authentic signer with probability better than $\frac{1}{n-t}$, where $n$ is the group size. Formally, let $D$ be a distinguisher who tries to break the anonymity of $\text{GS}$. Consider the following full anonymity experiment.

<table>
<thead>
<tr>
<th>Full Anonymity Experiment $\text{FA-Expt}_{D, \text{GS}}$:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Setup.</strong> The challenger $C$ initiates the system by running $(\text{PP}, msk, tsk) \leftarrow \text{Setup}(1^k)$. Public parameter $\text{PP}$ is given to $D$.</td>
</tr>
<tr>
<td><strong>Query 1.</strong> $D$ adaptively accesses to oracles $\text{OEnroll}, \text{OSig}, \text{OCorrupt}, \text{OTrace}$ and $\text{OLeak}$ for polynomially many times.</td>
</tr>
<tr>
<td><strong>Challenge.</strong> $D$ outputs a challenge message $m^<em>$ and two distinct challenge identifies $id_0, id_1$. $C$ tosses a random coin $b \xleftarrow{} {0, 1}$ and returns a challenge signature $\sigma^</em> \leftarrow \text{GSig}(\text{PP}, sk_{id_b}, m)$ to $D$.</td>
</tr>
<tr>
<td><strong>Query 2.</strong> $D$ is allowed to adaptively access to oracles $\text{OEnroll}, \text{OSig}, \text{OCorrupt}, \text{OTrace}$ and $\text{OLeak}$ for polynomially many times with restriction that $(m^<em>, \sigma^</em>) \notin \text{TList}$ must hold.</td>
</tr>
<tr>
<td><strong>Guess.</strong> Finally, $D$ outputs a bit $b'$ and wins iff. $b' = b$.</td>
</tr>
</tbody>
</table>

Anonymity with Bounded Leakage Resilience. We call a group signature scheme $\text{IBS}$ achieves $\lambda$-leakage resilience and full anonymity if the probability that any PPT distinguisher $D$ wins in the Full Anonymity Experiment $\text{FA-Expt}_{D, \text{IBS}}$ is

$$\Pr\left[b' = b : b' \leftarrow D^{\text{OEnroll}, \text{OSig}, \text{OCorrupt}, \text{OTrace}, \text{OLeak}}(\text{PP})\right] \leq \frac{1}{2} + \text{negl}(k).$$
**Full Traceability.** We say a group signature satisfies full traceability if no one, even collusive компромиссирован or group manager, can forge a signature that is traced to any honest group members. That is, it requires that it is computationally difficult for any PPT forger to produce a valid signature on a new message on behalf of an honest group member. It is formally described in the experiment as follows.

<table>
<thead>
<tr>
<th>Full Traceability Experiment FT-Expt,F,Gs:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Setup.</strong> The challenger $C$ initiates the system by running $(PP, msk, tsk) \leftarrow \text{Setup}(1^k)$. Public parameter $PP$ is given to $F$.</td>
</tr>
<tr>
<td><strong>Query</strong> $F$ adaptively accesses to oracles OEnroll, OSig, OCorrupt, OTrace and OLeak for polynomially many times.</td>
</tr>
<tr>
<td><strong>Challenge.</strong> Finally, $F$ outputs a forgery $(\hat{m}, \hat{\sigma})$, which is traced to identity $\hat{\text{id}} \leftarrow \text{Trace}(tsk, \hat{m}, \hat{\sigma})$. $F$ wins iff. (a) $\text{GVer}(PP, \hat{m}, \hat{\sigma}) = 1$, (b) $(\hat{\text{id}}, \hat{m}, \hat{\sigma}) \notin S\text{List}$, and (c) $\hat{\text{id}} \notin C\text{List}$.</td>
</tr>
</tbody>
</table>

$G$s achieves $\lambda$-leakage-resilient full anonymity if for any PPT forger $F$, the probability that $F$ wins in the Full Traceability Experiment FT-Expt,F,Gs, i.e.

$$\Pr \left[ \begin{array}{c} \text{GVer}(PP, \hat{m}, \hat{\sigma}) = 1 \\
(\hat{\text{id}}, \hat{m}, \hat{\sigma}) \notin S\text{List} \\
\hat{\text{id}} \notin C\text{List} \\
\end{array} \right] \leq \text{negl}(k).$$

**Full Leakage Resilience.** In the aforementioned full anonymity experiment and full traceability experiment, if the system state involves all signing keys of the group members as well as all randomness used in the signing process we say group signature scheme $G$s satisfies full leakage resilience.

4. Generic Constructions of Leakage-Resilient Group Signature

4.1. Construction 1: From Leakage-Resilient IBS and CCA-secure Encryption

**Intuition.** In this subsection, we present a black-box construction of leakage-resilient group signature scheme from a leakage-resilient public-key encryption scheme and a leakage-resilient IBS scheme. Let $\Pi = (Ek, Enc, Dec)$ be a public-key encryption scheme, $IBS = (Setup, Extract, Ib-Sig, Ib-Ver)$ be an identity-based signature scheme, and $\mathcal{NIZK} := (I, P, V, S = (S_1, S_2))$ be an unbounded simulation-sound NIZK proof argument for $NP$-language

$$L := \{ (mpk, m, ek, C) : \exists (id, \sigma), s.t. Ib-Ver(mpkm, id, m, \sigma) = 1 \land C = \Pi.Enc(ek, id||\sigma) \}.$$ 

Put things differently, an authenticated signer possesses a valid signing key to sign any messages. Therefore, the signer is able to generate a witness (valid signature) $\sigma$ on message $m$ and a ciphertext $C$ of the combination of his/her identity $id$ and the signature $\sigma$. Consider the following group signature scheme $\sigma_G = (Setup, Join, GSig, GVer, Trace)$. 

10
\textbf{Construction 1}

\textbf{Setup}(1^k). Given the security parameter 1^k, the system is initialized as follows.

1. Invoke \((mpk, msk) \leftarrow IBS.\text{Setup}(1^k)\) and \((ek, dk) \leftarrow \Pi.\text{EKg}(1^k)\).

2. Randomly pick a random string \(r \leftarrow \{0, 1\}^l(k)\).

3. Output the public parameter \(PP := (mpk, ek, r)\), the master key \(msk := msk\) and the tracing key \(tsk := dk\).

\textbf{Join}(\(PP, msk, id\)). Given the public parameter \(PP\), the master key \(msk\) and a user identity \(id\), return the corresponding user signing key \(sk_{id} \leftarrow IBS.\text{Extract}(msk, id)\) via secure channel.

\textbf{GSig}(\(PP, sk_{id}, m\)). Given a public parameter \(PP\), a signing key \(sk_{id}\) and a message \(m\), compute a group signature as follows.

1. Sign message \(m\) by invoking \(\sigma \leftarrow IBS.\text{Sig}(sk_{id}, m)\).

2. Encrypt identity \(id\) and signature \(\sigma\), i.e. \(C := \pi.\text{Enc}(ek, id || \sigma)\).

3. Generate the proof \(\pi \leftarrow \mathcal{P}(\text{mpk}, m, ek, C, (id, \sigma); r)\).

4. Output the signature \(\sigma_G := (C, \pi)\).

\textbf{GVer}(\(PP, m, \sigma_G\)). Given the public parameter \(PP\), a message \(m\) and a purported group signature \(\sigma_G := (C, \pi)\), the verification algorithm outputs \(V((mpk, m, ek, C), \pi; r)\).

\textbf{Trace}(\(PP, tsk, m, \sigma\)). Given the public parameter \(PP\), a tracing key \(tsk\), message \(m\) and signature \(\sigma_G\), if \(\text{GVer}(PP, m, \sigma_G) = 0\), return \(\perp\). Otherwise, compute \((id || \sigma) := \Pi.\text{Dec}(tsk, C)\). Return the corresponding binding identity \(id\).

\textbf{Security Analysis}.

\textbf{Theorem 1} If \(\Pi\) is CCA-secure and \(\lambda_1\)-leakage-resilient, \(IBS\) is UF-CMA secure with \(\lambda_2\)-bounded leakage resilience, and \(NIZK\) is an unbounded simulation-sound NIZK proof argument as described above, then group signature scheme \(\Sigma_G\) satisfies full anonymity and full traceability with \(\lambda\)-bounded leakage resilience where \(\lambda = \min(\lambda_1, \lambda_2)\).

\textbf{Proof (of Theorem 1)}. First we prove the full anonymity of \(\Sigma_G\). Consider the following hopping experiments and denote the probability that \(D\) outputs 0 in Full Anonymity Experiment \(i\) by \(p_i\).

\textbf{FA-Expt}_0. It is exactly the same as the full anonymity experiment in Sect. 3.3 with \(b = 0\). \(C \leftarrow \Pi.\text{Enc}(ek, id_0 || \sigma_0)\) where \(\sigma_0 \leftarrow \Sigma.\text{Sig}(sk_{id_0}, m^*)\). Then we have

\[
p_0 = \Pr[D_{\text{OEnroll}, \text{OSig}, \text{OCorrupt}, \text{OTrace}, \text{OLeak}(PP)}(b = 0)] = (1)
\]

\textbf{FA-Expt}_1. It is the same as Experiment 0 except that the common reference string \(r\) of the \(NIZK\) is generated by running \((r, \tau) \leftarrow S_1(1^k)\). Furthermore, to sign message \(m\), the proof \(\pi\) is now generated as
\[\pi \leftarrow S_2((mpk, m, ek, C), \tau).\] Following the (unbounded) zero-knowledge property of NIZK, we have
\[|p_1 - p_0| \leq \text{negl}(k).\] (2)

**FA-Expt_2.** It is the same as Experiment 1 except that in the challenge phase, to generate the challenge signature, compute \(C \leftarrow \Pi.E\text{nc}(ek, id_1||\sigma_1)\) and \(\sigma_1 \leftarrow \Sigma.S\text{ig}(sk_{id_1}, m^*)\). Then compute \(\pi\) same as in Experiment 1. CCA-security of the encryption scheme implies that
\[|p_2 - p_1| \leq \text{negl}(k).\] (3)

**FA-Expt_3.** It is the same as Experiment 2 except that in the challenge phase, we sample the random common string \(r \leftarrow \{0, 1\}^l\) instead of invoking simulation algorithm \(S\). Then, to sign a quired message \(m\), generate \(C\) same as in Experiment 2, but compute \(\pi\) by running \(P((vk, m, ek, C), (id_1, \sigma_1); r)\). From the zero-knowledge property of NIZK, we have
\[|p_3 - p_2| \leq \text{negl}(k).\] (4)

Furthermore, this experiment is exactly the same as full anonymity experiment in Sect. 3.3 with \(b = 1\). That is,
\[p_3 = Pr[0 \leftarrow \mathcal{D}^{O\text{Enroll}, OS\text{ig}, O\text{Corrupt}, O\text{Trace}, O\text{Leak}}(\mathcal{P})|b = 1].\] (5)

From eqs. (1) to (5), we have
\[Pr[b' = b : b' \leftarrow \mathcal{D}^{O\text{Enroll}, OS\text{ig}, O\text{Corrupt}, O\text{Trace}, O\text{Leak}}(\mathcal{P})] = p_0 \cdot Pr[b = 0] + (1 - p_3) \cdot Pr[b = 1]\]
\[= \frac{1}{2} + (p_0 - p_3) \cdot \frac{1}{2}\]
\[\leq \frac{1}{2} + \text{negl}(k).\]

This shows that \(\Sigma_G\) satisfies full anonymity property. Next, we prove the full traceability of \(\Sigma_G\). Assume that there exists a PPT forger \(\mathcal{F}\) that breaks the full traceability of \(\Sigma_G\) with non-negligible probability, then we can construct another PPT forger \(\mathcal{F}'\) that breaks the unforgeability of IBS with non-negligible probability as well. Let \(\epsilon\) be the probability of \(\mathcal{F}'\) wins in the UF-CMIA Experiment CMIA-ID\(A_{\mathcal{F},\text{IBS}}(k)\). Details are as follows.

**Algorithm \(\mathcal{F}'\).** Given a master public key \(mpk^*\) along with a user creation oracle \(O(\cdot)\), an extraction oracle \(E\mathcal{O}(\cdot)\), a signing oracle \(S\mathcal{O}(\cdot,\cdot)\) and a leakage oracle \(L\mathcal{O}(\cdot)\), \(\mathcal{F}'\) does as follows.

**Setup.** Sample random common reference string \(r \leftarrow \{0, 1\}^l(k)\) and set the leakage amount \(L = 0\). Invoke \((ek, dk) \leftarrow \Pi.E\text{Kg}(1^k)\). Set the public parameter \(\mathcal{P}P := (mpk^*, ek, r)\), the master key \(\text{msk} := \perp\) and the...
tracing key $tuk := dk$. The public parameter $PP$ is given to $F$. Initially set $EList = SList = CList = \emptyset$, $L = 0$ and $State := \{msk = \bot, tsk = dk\}$.

**Query.** Answer queries from $F$ as follows.

- **OEnroll.** On input an index of specific user identity $id$, generate a user signing key $sk_{id} \leftarrow EO(id)$. Update the enrollment query list $EList = EList \cup \{(id, sk_{id})\}$ and the system state $State := State \cup \{sk_{id}\}$.

- **OSig.** On input a specific user identity $id$ and a message $m$, retrieve the enrollment list to obtain the user signing key $sk_{id}$ s.t. $(id, sk_{id}) \in EList$. Compute $\sigma \leftarrow Sig(sk_{id}, m)$, $C := \Pi.Enc(ek, id || \sigma)$ and $\pi \leftarrow \mathcal{P}(mpk^*, m, ek, C), (id, \sigma); r)$. Update the signing query list $SList = SList \cup \{(id, m, \sigma)\}$.

- **OCorrupt.** On input a user identity $id$, return $sk_{id}$ and update the corruption query list $CList = CList \cup \{id\}$.

- **OTrace.** On input a valid message/signature pair $(m, (C, \pi))$, if $GVer(m, (C, \pi)) = 0$, output $\bot$. Otherwise, obtain $(id || \sigma) \leftarrow \Pi.Dec(tsk, C)$ and return $id$. Update the tracing query list $TList = TList \cup \{(id, m, (C, \pi))\}$.

- **OLeak.** On input a leakage function $f(\cdot)$, construct another equivalent leakage function $f'$ that embeds all terms of $State \setminus \{msk\}$ and issues a query $f'$ to $LO(\cdot)$. Return $\Lambda \leftarrow LO(f')$ and update $L = L + \Lambda$.

**Challenge.** Finally, $F$ outputs a forgery $(\hat{m}, (\hat{C}, \hat{\pi}))$. $F'$ opens $\hat{C}$ by running $(\hat{id}, \hat{\sigma}) \leftarrow \Pi.Dec(dk, \hat{C})$, and outputs $(\hat{id}, \hat{m}, \hat{\sigma})$.

Let $Succ$ be the event that the forgery $(\hat{C}, \hat{\pi})$ is a valid signature of message $\hat{m}$. This implies that (a) $(\hat{id}, \hat{m}, *) \not\in SList$, (b) $V((mpk^*, \hat{m}, ek, \hat{C}), \hat{\pi}; r)) = 1$, and (c) $\hat{id} \not\in CList$. Let $Ext$ be the event that $Succ$ occurs and furthermore, $GVer(mpk^*, \hat{id}, \hat{m}, \hat{\sigma}) = 1$ hold. Unbounded simulation soundness of the NIZK proof system implies that $|Pr[Succ] − Pr[Ext]| \leq \text{negl}(k)$. Then we have

\[
Pr[Succ] = (1 - \text{negl}(k)) \cdot Pr[Ext]
\]

\[
= (1 - \text{negl}(k)) \cdot Pr[GVer(mpk^*, \hat{id}, \hat{m}, \hat{\sigma}) = 1 \land O(\hat{id}, \hat{m}) \text{ is not quired} \land L(\hat{id}) \text{ is not quired}]
\]

\[
= (1 - \text{negl}(k)) \epsilon
\]

\[
= \text{negl'}(k)
\]

which results from the fact that $IBS$ is $\lambda_2$-leakage-resilient.

\[\square\]

**4.2. Construction 2: From Leakage-Resilient Signature and CCA-secure Encryption**

Let $\Pi = (EKg, Enc, Dec)$ be a public-key encryption scheme, $\Sigma = (Kg, Sig, Ver)$ be a standard existentially unforgeable signature scheme, and $\mathcal{NIZK} := (l, \mathcal{P}, V, S = (S_1, S_2))$ be an unbounded simulation-sound NIZK proof argument for $\mathcal{NP}$-language

\[
L := \left\{(vk, m, ek, C) : \exists(uvk, Cert, \sigma), s.t. \begin{array}{l}
\text{Ver}(vk, uvk, Cert) = 1 \land Ver(uvk, m, \sigma) = 1 \\
\land C = \Pi.Enc(ek, uvk || Cert || m || \sigma)
\end{array}\right\}.
\]

Consider the following group signature scheme $\sigma_G = (\text{Setup}, \text{Join}, \text{GSig}, \text{GVer}, \text{Trace})$. 

13
Construction 2

Setup(1^k). Given the security parameter 1^k, the system is initialized as follows.

1. Invoke (vk, sk) ← Σ.Kg(1^k).
2. (ek, dk) ← Π.EKg(1^k).
3. Randomly pick a random string r ← {0, 1}^{l(k)}.
4. Output the public parameter PP := (vk, ek, r), the master key msk := sk and the tracing key tsk := dk.

Join(PP, msk, id).

1. User with identity uid generates a verification/signing key pair by invoking (uvk, usk) ← Σ.Kg(1^k) and submit id = (uid, uvk) to group manager.
2. Group manager signs uid∥uvk with its master key msk, i.e. Cert ← Σ.Sig(msk, uid∥uvk).
3. Return Cert to user via public channel.
4. User sets the signing key skid := usk along with the group certificate signed by group manager.

GSig(PP, skid, m). Given a public parameter PP, a signing key skid along with a group certificate Cert and a message m, compute a group signature as follows.

1. Compute σ ← Σ.Sig(skid, m).
2. Compute C := π.Enc(ek, uvk||Cert||m||σ).
3. Generate the proof π ← P((vk, m, ek, C), (uvk, Cert, σ); r).
4. Output the signature σG := (C, π).

GVer(PP, m, σG). Given a public parameter PP, a message m and a purported group signature σG := (C, π), the verification algorithm outputs V((vk, m, ek, C), m; π, r).

Trace(PP, tsk, m, σ). Given the public parameter PP, a tracing key tsk, message m and signature σG, if GVer(PP, m, σG) = 0, return ⊥. Otherwise, compute (uvk||Cert||m||σ) ← Π.Dec(tsk, C). Return the corresponding binding identity id.

Theorem 2 If Π is CCA-secure and λ_1-leakage-resilient, Σ is UF-CMA secure with λ_2-bound leakage resilience, and NIZK is an unbounded simulation-sound NIZK proof argument as described above, then group signature scheme Σ_G satisfies full anonymity and full traceability with λ-bound leakage resilience where λ = min(λ_1, λ_2).

Proof (of Theorem 2). First we prove the full anonymity of Σ_G. Consider the following hopping experiments and denote the probability that D outputs 0 in Full Anonymity Experiment i by p_i.

FA-Expt_i. It is exactly the same as the full anonymity experiment in Sect. 3.3 with b = 0. That is,
\[ C \leftarrow \Pi.\text{Enc}(ek, uvk_0 \| Cert_0 \| m^* \| \sigma_0) \] where \( Cert_0 \leftarrow \Sigma.Kg(msk, id_0) \) and \( \sigma_0 \leftarrow \Sigma.Sig(usk_0, m^*) \). Then we have

\[ p_0 = \Pr[b \leftarrow D^{OEnroll,OSig,OCorrupt,OTrace,OLeak}(PP)|b = 0]. \]  

(6)

**FA-Expt\(_1\)**. It is the same as Experiment 0 except that the common reference string \( r \) of the NIZK is generated by running \((r, \tau) \leftarrow S_1(1^k)\). Furthermore, to sign message \( m^* \), the proof \( \pi \) is now generated as \( \pi \leftarrow S_2((vk, m^*, ek, C), \tau) \). Following the (unbounded) zero-knowledge property of NIZK, we have

\[ |p_1 - p_0| \leq \text{negl}(k). \]  

(7)

**FA-Expt\(_2\)**. It is the same as Experiment 1 except that in the challenge phase, to generate the challenge signature, compute \( C \leftarrow \Pi.\text{Enc}(ek, uvk_1 \| Cert_1 \| m^* \| \sigma_1) \) where \( Cert_1 \leftarrow \Sigma.Kg(msk, id_1) \) and \( \sigma_1 \leftarrow \Sigma.Sig(usk_1, m^*) \). Then compute \( \pi \) same as in Experiment 1. CCA-security of the encryption scheme implies that

\[ |p_2 - p_1| \leq \text{negl}(k). \]  

(8)

**FA-Expt\(_3\)**. It is the same as Experiment 2 except that in the challenge phase, we sample the random common string \( r \leftarrow \{0, 1\}^l \) instead of invoking simulation algorithm \( S \). Then, to sign a queried message \( m \), generate \( C \) same as in Experiment 2, but compute \( \pi \) by running \( P((vk, m, ek, C), (uvk, Cert, \sigma); r) \). From the zero-knowledge property of NIZK, we have

\[ |p_3 - p_2| \leq \text{negl}(k). \]  

(9)

Furthermore, this experiment is exactly the same as full anonymity experiment in Sect. 3.3 with \( b = 1 \). That is,

\[ p_3 = \Pr[b' \leftarrow D^{OEnroll,OSig,OCorrupt,OTrace,OLeak}(PP)|b' = 1]. \]  

(10)

From eqs. (6) to (10), we have

\[
\Pr[b' = b : b' \leftarrow D^{OEnroll,OSig,OCorrupt,OTrace,OLeak}(PP)]
\]

\[ = p_0 \cdot \Pr[b = 0] + (1 - p_3) \cdot \Pr[b = 1]
\]

\[ = \frac{1}{2} + (p_0 - p_3) \cdot \frac{1}{2}
\]

\[ \leq \frac{1}{2} + \text{negl}(k).
\]

Therefore, \( \Sigma_G \) satisfies full anonymity property.

Next, we prove the full traceability of \( \Sigma_G \). Assume that there exists a PPT forger \( \mathcal{F} \) that breaks the full traceability of \( \Sigma_G \) with non-negligible probability, then we can construct another PPT forger \( \mathcal{F}' \) that breaks the unforgeability of \( \Sigma \) with non-negligible probability as
well. We consider following two types of forgers, Type-I forger $\mathcal{F}_1'$ and Type-II forger $\mathcal{F}_{II}'$ who succeed in forging a group signature.

**Type-I.** The valid signature is traced to a signer $id^*$ who is not a group member. That is, $(id^*, \ast) \notin EList$, which implies that $\mathcal{F}_1'$ succeeds in forging a new group certificate (signature) on $id^*$, which breaks the unforgeability of $\Sigma$ on master signing key $msk$.

**Type-II.** The valid signature is traced to an honest signer $id^*$. In other words, the signing key $O\text{Corrupt}(id^*)$ is not quired before and $\mathcal{F}_{II}'$ succeeds in forging a signature $\sigma^*$ on new message $m^*$, which breaks the unforgeability of $\Sigma$ on user signing key $usk$.

Let $\epsilon_1, \epsilon_2$ respectively be the probabilities of $\mathcal{F}_1'$ and $\mathcal{F}_{II}'$ wins in the UF-CMIA Experiment. Details are as follows.

**Algorithm $\mathcal{F}_1'$.** Given a verification key $vk^*$ along with a signing oracle $O_S(\cdot)$ and a leakage oracle $O_L(\cdot)$, $\mathcal{F}_1'$ does as follows.

**Setup.** Sample random common reference string $r \leftarrow \{0, 1\}^l(k)$ and set the leakage amount $L = 0$. Invoke $(ek, dk) \leftarrow II.EKg(1^k)$. Set the public parameter $PP := (vk^*, ek, r)$, the master key $msk := \bot$ and the tracing key $tsk := dk$. The public parameter $PP$ is given to $\mathcal{F}$. Initially set $EList = SList = CList = \phi$, $L = 0$ and $State := \{msk = \bot, tsk = dk\}$.

**Query.** Answer queries from $\mathcal{F}$ as follows.

**OEnroll.** On input an index of specific user identity $uid$, generate a user signing key by running $(usk, uvk) \leftarrow \Sigma.Kg(1^k)$ and obtain $Cert \leftarrow O_S(uid\|uvk)$. Update the enrollment query list $EList = EList \cup \{(uid, uvk, usk, Cert)\}$ and the system state $State := State \cup \{usk\}$.

**OSig.** On input a specific user identity $id := (uid, uvk)$ and a message $m$, retrieve the enrollment list to obtain the user signing key $sk_{id} := (usk, Cert)$ s.t. $(uid, uvk, usk, Cert) \in EList$. Compute $\sigma \leftarrow \Sigma.Sig(usk, m, C := II.Enc(ek, uvk\|Cert\|m\|\sigma))$ and $\pi \leftarrow \mathcal{P}((vk^*, m, ek, C), (uvk, Cert, \sigma); r)$. Update the signing query list $SL = SList \cup \{(id := (uid, uvk), \sigma\)}$.

**OCorrupt.** On input a user identity $id$, return $sk_{id}$ and update the corruption query list $CL = CList \cup \{id\}$.

**OTrace.** On input a valid message/signature pair $(m, (C, \pi))$, if $GVer(m, (C, \pi)) = 0$, output $\bot$. Otherwise, obtain $uvk\|Cert\|m\|\sigma \leftarrow II.Dec(tsk, C)$, retrieve enrollment list to find $uid$ s.t. $(uid, uvk, *, *) \in EList$ and return $uid$. Update the tracing query list $TList = TList \cup \{(id, m, (C, \pi))\}$.

**OLeak.** On input a leakage function $f(\cdot)$, construct another equivalent leakage function $f'$ that embeds all terms of $State \setminus \{msk\}$ and issues a query $f'$ to $LO(\cdot)$. Return $\Lambda \leftarrow LO(f')$ and update $L = L + \Lambda$.

**Challenge.** Finally, $\mathcal{F}$ outputs a forgery $(\hat{m}, (\hat{C}, \hat{\pi}))$. $\mathcal{F}_1'$ opens $\hat{C}$ by running $(\hat{id}, \hat{\sigma}) \leftarrow II.Dec(dk, \hat{C})$, and outputs $(\hat{uvk}, \hat{Cert})$.

Let $\text{Succ}$ be the event that the forgery $(\hat{C}, \hat{\pi})$ is a valid signature of message $\hat{m}$. Again, forger $\mathcal{F}$ is able to produce a valid signature on a new message that is traced to a non-existent group member. This implies
that (a) \((*, \hat{u}vK) \notin \mathcal{E}List\), (b) \((*, \hat{u}vK), \hat{m}, *) \notin \mathcal{S}List\), (c) \(\forall((vk^*, \hat{m}, ek, \hat{C}), \hat{r}; r)) = 1\), and (d) \(\hat{u}vK \notin \mathcal{C}List\).

Let \(\mathcal{E}xt\) be the event that \(\mathcal{S}ucc\) occurs and furthermore, \(\mathcal{V}er(vk^*, \hat{u}vK, \hat{m}, \hat{C}ert) = 1\) and \(\mathcal{V}er(\hat{u}vK, \hat{m}, \hat{S}) = 1\) hold. Unbounded simulation soundness of the NIZK proof system implies that \(|\Pr[\mathcal{S}ucc]−\Pr[\mathcal{E}xt]| \leq \negl(k)\).

Then we have

\[
\Pr[\mathcal{S}ucc] = (1−\negl(k)) \cdot \Pr[\mathcal{E}xt]
\]

\[
= (1−\negl(k)) \cdot \Pr[\mathcal{V}er(vk^*, \hat{m}, \hat{C}ert) = 1 \land O_\lambda(\hat{u}vK) \text{ is not required}]
\]

\[
= (1−\negl(k))c_1
\]

\[
= \negl'(k)
\]

which results from the fact that \(\Sigma\) is \(\lambda_2\)-leakage-resilient. Next, we consider forger \(\mathcal{F}_H'\) as follows.

**Algorithm \(\mathcal{F}_H'\).** Given a verification key \(vk^*\) along with a signing oracle \(O_\lambda(\cdot)\) and a leakage oracle \(O_L(\cdot)\), \(\mathcal{F}_H'\) does as follows.

**Setup.** Sample random common reference string \(r \xleftarrow{\$} \{0,1\}^l(k)\) and set the leakage amount \(L = 0\). Invoke \((vk, sk) \leftarrow \Sigma.Kg(1^k)\) and \((ek, dk) \leftarrow \Pi.EKg(1^k)\). Set the public parameter \(PP := (ek, ek, r)\), the master key \(msk := sk\) and the tracing key \(tsk := dk\). The public parameter \(PP\) is given to \(\mathcal{F}\). Randomly pick \(i^* \leftarrow [n]\) where \(n := n(k)\) is the group size. Initially set \(\mathcal{E}List = \mathcal{S}List = \mathcal{C}List = \phi\), \(L = 0\) and \(\text{State} := \{msk = sk, tsk = dk\}\).

**Query.** Answer queries from \(\mathcal{F}\) as follows.

\(\mathcal{O}_{\text{Enroll}}\). Denote the \(i\)th user identity by \(uid_i\). Obtain a verification/signing key pair

\[
(\hat{u}vK_i, usk_i) := \begin{cases} 
\Sigma.Kg(1^k) & i \neq i^*
\end{cases}
\]

and generate the group certificate by running \(\mathcal{C}ert \leftarrow \Sigma.Sg(msk, uid_i || \hat{u}vK_i)\). Update the enrollment query list \(\mathcal{E}List = \mathcal{E}List \cup \{(uid_i, \hat{u}vK_i, usk_i, \mathcal{C}ert_i)\}\) and the system state \(\text{State} := \text{State} \cup \{usk_i\}\). Notice that \(usk_i\) is unknown.

\(\mathcal{O}_{\text{Sig}}\). On input a specific user identity \(id := (uid, \hat{u}vK)\) and a message \(m\), retrieve the enrollment list to obtain the user signing key \(sk_{id} := (usk, \mathcal{C}ert)\) s.t. \((uid, \hat{u}vK, usk, \mathcal{C}ert) \in \mathcal{E}List\). Compute \(C := \Pi.Enc(ek, \hat{u}vK || \mathcal{C}ert || m || \sigma)\) and \(\pi \leftarrow \mathcal{P}((vk, m, ek, C), (\hat{u}vK, \mathcal{C}ert, \sigma); r)\) where

\[
\sigma := \begin{cases} 
\Sigma.Sg(usk, m) & \hat{u}vK \neq vk^*
\end{cases}
\]

\[
O_\lambda(m) & \hat{u}vK = vk^*
\]

Update the signing query list \(\mathcal{S}List = \mathcal{S}List \cup \{(id := (uid, \hat{u}vK), m, \sigma)\}\).

\(\mathcal{O}_{\text{Corrupt}}\). On input a user identity \(id := (uid, \hat{u}vK)\), if \(\hat{u}vK = vk^*\), abort. Otherwise, return \(sk_{id}\) and update the corruption query list \(\mathcal{C}List = \mathcal{C}List \cup \{id\}\).
Then we have

\[ \Pi \text{ is language-unforgeable.} \]

4.3. Construction 3: From Leakage-Resilient Signature and CPA-Secure Encryption

Let \( (\Pi = (\Pi_{\text{Kg}}, \Pi_{\text{Enc}}, \Pi_{\text{Dec}})) \) be a public-key encryption scheme, \( \Sigma = (\Pi_{\text{Kg}}, \Pi_{\text{Sig}}, \Pi_{\text{Ver}})) \) be a standard existentially unforgeable signature scheme, and \( \mathcal{NIZK} := (l, \mathcal{P}, \mathcal{V}, \mathcal{S} = (S_1, S_2)) \) be an unbounded simulation-sound NIZK proof argument for \( \mathcal{NP} \)-language

\[ L := \left\{ (v_k, m, \nu_k, C_1, C_2) : \exists (\nu_k, C_1, m, \sigma \mid \omega_1, \omega_2), \text{s.t.} \begin{align*}
\Pi_{\text{Ver}}(v_k, \nu_k, C_1) &= 1 \\
\Pi_{\text{Ver}}(v_k, \nu_k, C_1) &= 1
\end{align*}ight\} \]

Consider the following group signature scheme \( \sigma_G = (\Pi_{\text{Setup}}, \Pi_{\text{Join}}, \Pi_{\text{GSig}}, \Pi_{\text{GVer}}, \Pi_{\text{Trace}}) \).
Construction 3

Setup$(1^k)$. Given the security parameter $1^k$, the system is initialized as follows.

1. Invoke $(vk, sk) \leftarrow \Sigma.Kg(1^k)$.
2. $(ek_1, dk_1) \leftarrow \Pi.EKg(1^k)$ and $(ek_2, dk_2) \leftarrow \Pi.EKg(1^k)$.
3. Randomly pick a random string $r \gets \{0, 1\}^{(k)}$.
4. Output the public parameter $PP := (vk, ek_1, ek_2, r)$, the master key $msk := sk$ and the tracing key $tsk := dk_1$.

Join$(PP, msk, id)$. 

1. User with identity $uid$ generates a verification/signing key pair by invoking $(uvk, usk) \leftarrow \Sigma.Kg(1^k)$ and submit $id = (uid, uvk)$ to group manager.
2. Group manager signs $uid||uvk$ with its master key $msk$, i.e. $Cert \leftarrow \Sigma.Sig(msk, uid||uvk)$.
3. Return $Cert$ to user via public channel.
4. User sets the signing key $sk_{id} := usk$ along with the group certificate signed by group manager.

GSig$(PP, sk_{id}, m)$. Given a public parameter $PP$, a signing key $sk_{id}$ along with a group certificate $Cert$ and a message $m$, compute a group signature as follows.

1. Compute $\sigma \leftarrow \Sigma.Sig(sk_{id}, m)$.
2. Compute $C_1 := \pi.\text{Enc}(ek_1, uvk||Cert||m||\sigma; \omega_1)$ and $C_2 := \pi.\text{Enc}(ek_2, uvk||Cert||m||\sigma; \omega_2)$.
3. Generate the proof $\pi \leftarrow \mathcal{P}((vk, m, ek_1, ek_2, C_1, C_2), (uvk, Cert, \sigma, \omega_1, \omega_2); r)$.
4. Output the signature $\sigma_G := (C, \pi)$.

GVer$(PP, m, \sigma_G)$. Given a public parameter $PP$, a message $m$ and a purported group signature $\sigma_G := (C, \pi)$, the verification algorithm outputs $V((vk, m, ek_1, ek_2, C), \pi; r)$.

Trace$(PP, tsk, m, \sigma)$. Given the public parameter $PP$, a tracing key $tsk$, message $m$ and signature $\sigma_G$, if $\text{GVer}(PP, m, \sigma_G) = 0$, return $\bot$. Otherwise, compute $(uvk||Cert||m||\sigma) \leftarrow \Pi.\text{Dec}(tsk, C)$. Return the corresponding binding identity $id$.

Theorem 3 If $\Pi$ is CPA-secure and $\lambda_1$-leakage-resilient, $\Sigma$ is UF-CMA secure with $\lambda_2$-bounded leakage resilience, and $NI\text{ZK}$ is an unbounded simulation-sound NIZK proof argument as described above, then group signature scheme $\Sigma_G$ satisfies full anonymity and full traceability with $\lambda$-bounded leakage resilience where $\lambda = \min(\lambda_1, \lambda_2)$.

The proof of theorem 3 is similar to that of theorem 2 and thus we omit it here. The main difference is that the way to answer tracing queries from the adversary. We are able to correctly simulate the tracing process if the decryption oracle of CCA-secure encryption scheme is given. For a CPA-secure encryption,
we generate two uniform key pairs \((ek_1,dk_1)\) and \((ek_2,dk_2)\) where \(dk_2\) is erased at once in the setup process. In the simulation, the tracing key \(dk_1\) is unknown. To answer tracing query, we do not erase \(dk_2\) while the view of forger being not changed. Therefore, we are able to correctly to simulate the tracing query.

5. Conclusion

In this work we formalize the definition of leakage-resilient group signature scheme. Based on this model, we present three black-box constructions of group signature scheme. They are constructed from constructing from the combination of an IBS scheme and a CCA-secure PKE scheme, the combination of a signature scheme and a CCA-secure PKE scheme, the combination of a signature scheme and a CPA-secure PKE scheme, respectively. We prove that the leakage bound of resulting construction is the same as the lowest leakage bound of underlying primitives. Intuitively, the buckets effect is inescapable when the private state generated from different cryptographic primitives and the weakest part of the system is always more vulnerable.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 61472146, 61872152), Guangdong Natural Science Funds for Distinguished Young Scholar (No. 2014A030306021), Guangdong Program for Special Support of Top-notch Young Professionals (No. 2015TQ01X796), and Pearl River Nova Program of Guangzhou (No. 201610010037).

References


[40] Craig Ramsay. TEMPEST attacks against AES Covertly stealing keys for €200, 2017.


