University of Wollongong

Research Online

Faculty of Engineering and Information
Sciences - Papers: Part A

Faculty of Engineering and Information
Sciences

2013

Identity-based multisignature with message recovery

Kefeng Wang

University of Wollongong, kw909@uowmail.edu.au

Yi Mu

University of Wollongong, ymu@uow.edu.au

Willy Susilo

University of Wollongong, wsusilo@uow.edu.au

Follow this and additional works at: <https://ro.uow.edu.au/eispapers>

Part of the [Engineering Commons](#), and the [Science and Technology Studies Commons](#)

Research Online is the open access institutional repository for the University of Wollongong. For further information contact the UOW Library: research-pubs@uow.edu.au

Identity-based multisignature with message recovery

Abstract

We present a new notion of short identity-based multisignature scheme with message recovery. We propose a concrete identity-based multisignature with message recovery scheme based on bilinear pairing in which multiple signers can generate a constant size multisignature on same message regardless of the number of signers. There is no requirement to transmit the original message to the verifier, since the original message can be recovered from the multisignature. Therefore, this scheme minimizes the total length of the original message and the appended multisignature. The proposed scheme is proven to be existentially unforgeable against adaptively chosen message attacks in the random oracle model under the assumption that the Computational Diffie-Hellman problem is hard.

Keywords

recovery, message, identity, multisignature

Disciplines

Engineering | Science and Technology Studies

Publication Details

Wang, K., Mu, Y. & Susilo, W. (2013). Identity-based multisignature with message recovery. *Lecture Notes in Computer Science*, 7863 91-104.

Identity-based Multisignature with Message Recovery

Kefeng Wang, Yi Mu, and Willy Susilo*

Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
University of Wollongong, Wollongong NSW 2522, Australia
{kw909, ymu, wsusilo}@uow.edu.au

Abstract. We present a new notion of short identity-based multisignature scheme with message recovery. We propose a concrete identity-based multisignature with message recovery scheme based on bilinear pairing in which multiple signers can generate a constant size multisignature on same message regardless of the number of signers. There is no requirement to transmit the original message to the verifier, since the original message can be recovered from the multisignature. Therefore, this scheme minimizes the total length of the original message and the appended multisignature. The proposed scheme is proven to be existentially unforgeable against adaptively chosen message attacks in the random oracle model under the assumption that the Computational Diffie-Hellman problem is hard.

Keywords: Multisignature, Message Recovery, ID-based Cryptography.

1 Introduction

In networks with limited bandwidth and lightweight mobile devices, long digital signatures will obviously be a drawback. Apart from shortening the signature itself, the other effective approach for saving bandwidth is to eliminate the need to transmit the signed original message for verifying a digital signature. In this work, we consider on the latter approach.

Consider n different signers. In order to allow any subgroup of them to produce a joint signature on a message m and convince a verifier that each member of the stated subgroup signed the message, two or more signers cooperate to generate a single compact digital signature in a multisignature scheme. A single multisignature can greatly save communication costs instead of transmitting several individual signatures. To verify the validity of a multisignature, one still needs public keys of all signers. In most applications these public keys will have to be transmitted along with the multisignature. In this case, it partially defeats the primary purpose of using a multisignature scheme, namely to save bandwidth. But the inclusion of some information that uniquely identifies the signers

* This work is supported by the ARC Future Fellowship (FT0991397).

seems inevitable for verification. Fortunately, in an identity-based setting, this information can be represented in a more succinct way.

An identity-based signature scheme allows any pair of users to verify each other's signatures without exchanging public key certificates. It resembles an ideal mail system: If you know somebody's name and address you can send him messages that only he can read, and you can verify the signatures that only he could have produced. Compared to the public key of the signer is essentially a random bit string picked from a given set in traditional public key signature algorithms, in the identity-based scenario, the public key of a signer is simply his identity such as his name, email or IP address. The associated private key can only be computed by a trusted Private Key Generator (PKG) using a master secret. It can avoid using certificates which is a big burden to bandwidth in the verifying process of a signature. These features make the identity-based concept particularly appealing for use in conjunction with multisignatures.

When bandwidth is at a premium, another potential problem is that the combined length of the original message and the signature is too large. Signature schemes with total or partial message recovery provide a solution to this problem by embedding all or part of the message within the signature itself. That is, the message does not need to be hashed or sent along with the signature, which saves storage space and communication bandwidth.

Our Contributions. For the first time, this paper presents a provably secure (existentially unforgeable against adaptively chosen message attacks) identity-based multisignature with message recovery scheme based on bilinear pairing under the Computational Diffie-Hellman assumption in the random oracle model. Because the original message can be recovered from the multisignature, there is no need to transmit the original message to the verifier. This scheme minimizes the total length of the original message and the multisignature. We also present a concrete analysis of the reduction to prove the security of the proposed multisignature scheme. More precisely, we can show that if there is an attacker who can forge a valid multisignature to pass the verification, then the Computational Diffie-Hellman problem is solved.

Paper Organization. The rest of this paper is organized as follows: In Section 2, we introduce some related works that have been studied in the literature. In Section 3, we introduce some preliminaries used throughout this paper. In Section 4, we propose a notion of identity-based multisignature with message recovery scheme and present a concrete scheme based on bilinear pairing. We also present a security model and security proof of our scheme in this section. Section 5 concludes the paper.

2 Related Works

In 1984, Shamir introduced the notion of identity-based cryptography to simplify key management of certificate-based public key infrastructures and proposed an identity-based signature scheme [13]. Since then several practical identity-based

signature schemes have been devised [4, 6, 3, 8]. Cha and Cheon [3] proposed an identity-based signature scheme using gap Diffie-Hellman (GDH) groups, and proved their scheme is secure against existential forgery on adaptively chosen message and ID attack under the random oracle model. Hess [8] also proposed an efficient identity-based signature scheme based on pairings. The security of their scheme relies on the hardness of the Diffie-Hellman problem in the random oracle model.

The notion of multisignatures was introduced by Itakura and Nakamura [9]. Several works on this topic have been done [2, 10, 12]. In [10], the first formalized strong notion of security for multisignatures was proposed. They modified the Schnorr-signature-based multisignature scheme originally proposed by Ohta and Okamoto [12] and proved its security. Gangishetti et al. [5] presented identity-based serial and parallel multisignature schemes using bilinear pairings. Harn and Ren [7] proposed an efficient RSA multisignature scheme based on Shamir's identity-based signature.

In order to minimize the total length of the original message and the appended signature, the message recovery schemes were introduced (e.g. [11]). Zhang et al. [14] proposed an identity-based message recovery signatures scheme. Their scheme can be regarded as the identity based version of Abe-Okamoto's scheme [1]. Their scheme was also extended to achieve an identity-based partial message recovery signature scheme. Based on the scheme due to Zhang et al. [14], we achieved the goal of minimizing the total length of the original message and the appended multisignature in an identity-based setting.

3 Preliminaries

3.1 Bilinear Pairing

Let $\mathbb{G}_1, \mathbb{G}'_1$ be cyclic additive groups generated by P_1, P'_1 , respectively, whose order are a prime q . Let $\mathbb{G}_2$ be a cyclic multiplicative group with the same order q . We assume there is an isomorphism $\psi : \mathbb{G}'_1 \rightarrow \mathbb{G}_1$ such that $\psi(P'_1) = P_1$. Let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}'_1 \rightarrow \mathbb{G}_2$ be a bilinear mapping with the following properties:

- Bilinearity: $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$ for all $P \in \mathbb{G}_1, Q \in \mathbb{G}'_1, a, b \in \mathbb{Z}_q$.
- Non-degeneracy: There exists $P \in \mathbb{G}_1, Q \in \mathbb{G}'_1$ such that $\hat{e}(P, Q) \neq 1$.
- Computability: There exists an efficient algorithm to compute $\hat{e}(P, Q)$ for all $P \in \mathbb{G}_1, Q \in \mathbb{G}'_1$.

For simplicity, hereafter, we set $\mathbb{G}_1 = \mathbb{G}'_1$ and $P_1 = P'_1$. We note that our scheme can be easily modified for a general case, when $\mathbb{G}_1 \neq \mathbb{G}'_1$.

3.2 CDH Problem

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two groups of order the same prime order q . Let P be a generator of $\mathbb{G}_1$. Suppose there exists a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. Let $\mathcal{A}$ be an attacker. $\mathcal{A}$ tries to solve the following problem: Given (P, aP, bP) for some unknown $a, b \in \mathbb{Z}_q^*$, compute abP .

The success probability of $\mathcal{A}$, which is polynomially bounded with a security parameter l , is defined as

$$Succ_{\mathbb{G}_1, \mathcal{A}}^{CDH}(l) = \Pr[\mathcal{A}(P, aP, bP, abP) = 1; a, b \in \mathbb{Z}_q^*]$$

The CDH problem is said to be intractable, if for every probabilistic polynomial time algorithm $\mathcal{A}$, $Succ_{\mathbb{G}_1, \mathcal{A}}^{CDH}(l)$ is negligible.

4 Identity-based Multisignature with Message Recovery

4.1 Definitions

In an identity-based multisignature with message recovery scheme, there is a trusted party Private Key Generator (PKG). PKG is required to generate all the users' private keys.

There are three parties in the system, the PKG, the signer and the verifier. The scheme is ideal for closed groups of users such as the executives of a multinational company or the branches of a large bank, since the headquarters of the corporation can serve as a key generation center that everyone trusts. This scheme consists of the following four algorithms.

Setup: PKG sets up its secret key s with respect to a security parameter q as the master key of this scheme and publishes the corresponding public key P_{pub} . PKG should generate related groups and point out the generator of these groups. PKG also should describe which bilinear mapping and hash functions will be used in this scheme and publish these public information to all interested principals.

Extract: When a principal requires its private key S_{ID} corresponding its identity ID , this algorithm generates the private key using the master key and the principal's identity, and returns the private key to the principal.

Sign: This is an interactive algorithm. Several principals who got their private keys from the Extract algorithm can firstly generate their individual signatures (v_i, r, U_i) on a message m respectively, and one of them or other specified trusted principal can generate a single compact multisignature (r, U) on the message m corresponding to these principals who participate in this algorithm.

Verify: On receiving a multisignature (r, U) and several principal's identities $ID_1, ID_2, \dots, ID_n$, this algorithm checks whether the multisignature is a valid multisignature corresponding to these principal's public keys. If the multisignature is checked as valid, the original message m can be recovered from this multisignature.

4.2 Security Model

Boldyreva [2] defined the notion of security for multisignature as no valid multisignature should keep an honest player that part of the alleged subgroup accountable if it did not participate in signing. That is to say, no adversary can

forge an alleged multisignature of some message corresponding to an alleged subgroup of signers so that a verifier can check the multisignature as valid when not all signers of the alleged subgroup did sign the message. In order to achieve its goal, an adversary is allowed to corrupt players and send arbitrary messages during multisignature generation process.

We use a similar definition of existential unforgeability against a chosen message attack of [2]. Our definition is strong enough to capture an adversary who can simulate and observe the scheme. It is defined using the following game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

Assume in a subgroup of n signers who want to participate in generating a multisignature, there is only one honest signer. All other $n - 1$ members of the subgroup have been corrupted by the adversary. This means the adversary can get secret keys and public keys of corrupted signers. But the adversary only knows the public key of the single honest signer. The adversary can participate in the multisignature generation process on behalf of these $n - 1$ corrupted signers. Its goal is to frame the honest signer.

Firstly, challenger $\mathcal{C}$ runs Setup algorithm to get the system's master-key s with respect to a security parameter l and sends the system's public key $P_{pub} = sP$ and other public parameters $\{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, P, H_1, H_2, F_1, F_2, k_1, k_2\}$ to adversary $\mathcal{A}$.

$\mathcal{A}$ can access the following oracles to start an attack.

H₁ Oracle: For each H_1 hash query with respect to elements $v_1, v_2, \dots, v_n$ in $\mathbb{G}_2$ and a message m , $\mathcal{C}$ returns a hash value $H_1(v) \in_R \mathbb{Z}_q^*$ corresponding to the product $v = \prod_{i=1}^n v_i$ of these elements $v_1, v_2, \dots, v_n$.

H₂ Oracle: For each H_2 hash query with respect to an user ID_i , $\mathcal{C}$ returns a hash value $Q_{ID_i} \in_R \mathbb{G}_1$ as the user ID_i 's public key.

Extract Oracle: For each Extract query with respect to a user ID_i except for the honest user ID^* , $\mathcal{C}$ returns $S_{ID_i} = sQ_{ID_i}$ as the user's private key, in which the Q_{ID_i} is the H_2 hash value of the user ID_i 's identity.

Sign Oracle: For each Sign query on arbitrary message m with respect to a subgroup of n signer's identities $ID_1, ID_2, \dots, ID_n$, this oracle can be divided into two phases.

In the first phase, $n - 1$ signers generate their individual v_i by randomly selecting an element K_i from $\mathbb{G}_1$ and then computing $v_i = \hat{e}(K_i, P)$. These $n - 1$ signers send their v_i and a target signer's identity ID_t to $\mathcal{C}$. $\mathcal{C}$ outputs a random element $v_t \in_R \mathbb{G}_2$ corresponding to the target signer ID_t .

In the second phase, these $n - 1$ signers compute v using v_t and all v_i as $v = \prod_{i=1}^{n-1} v_i \cdot v_t$. At the same time, $\mathcal{C}$ computes the same v using the same method. These $n - 1$ signers generate and send their own individual signatures (v_i, r, U_i) and message m to $\mathcal{C}$. $\mathcal{C}$ returns a valid multisignature (r, U) on message m with respect to n signers include these $n - 1$ signers and the target signer.

Output: $\mathcal{A}$ outputs an alleged multisignature (r, U) on a target message m^* with respect to a subgroup of n signers $ID_1, \dots, ID^*, \dots, ID_n$ in which includes

an honest signer ID^* who did not participate in the multisignature generation process. If there was no Sign queries with respect to the target message m^* and a subgroup of signers in which includes the honest signer ID^* have been queried to Sign Oracle, and there was no Extract query with respect to the honest signer ID^* has been queried to Extract Oracle, $\mathcal{A}$ wins the game if the multisignature (r, U) can be verified as a valid multisignature.

If there is no such polynomial-time adversary that can forge a valid multisignature with respect to a subgroup of signers which includes an honest signer, while the honest signer did not participate in the multisignature generation process in the game described above, we say that the multisignature scheme is secure against existential forgery under chosen message attack.

The success probability of an adversary to win the game is defined by

$$Succ_{\mathcal{A}}^{UF-IDMMR-CMA}(l).$$

We say that an identity-based multisignature with message recovery scheme is existentially unforgeable under a chosen message attack if the success probability of any polynomially bounded adversary in the above game is negligible. In other words,

$$Succ_{\mathcal{A}}^{UF-IDMMR-CMA}(l) \leq \epsilon.$$

4.3 Proposed Scheme

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two groups of the same prime order q . Let P be a generator of $\mathbb{G}_1$. Suppose there exists a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$.

Setup: PKG chooses a random number $s \in \mathbb{Z}_q^*$ and keeps it as the master-key of this system. This master-key is known only by PKG itself. PKG sets $P_{pub} = sP$ as the system's public key and publishes this public key and other system parameters $\{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, P, H_1, H_2, F_1, F_2, k_1, k_2\}$.

Here $|q| = k_1 + k_2$. $H_1 : \{0, 1\}^* \rightarrow \mathbb{Z}_q^*$, $H_2 : \{0, 1\}^* \rightarrow \mathbb{G}_1^*$, $F_1 : \{0, 1\}^{k_2} \rightarrow \{0, 1\}^{k_1}$ and $F_2 : \{0, 1\}^{k_1} \rightarrow \{0, 1\}^{k_2}$ are four cryptographic hash functions.

Extract: A user submits his/her identity information ID_i to PKG. PKG computes the user's public key as $Q_{ID_i} = H_2(ID_i)$, and returns $S_{ID_i} = sQ_{ID_i}$ to the user as his/her private key.

Sign: Let the message be $m \in \{0, 1\}^{k_2}$.

Each signer randomly selects an element K_i in $\mathbb{G}_1$ and computes $v_i = \hat{e}(K_i, P)$. v_i is broadcast to other signers.

Once each signer's v_i are available through the broadcast channel. They compute their individual signatures as follows:

$$\begin{aligned} v &= \prod_{i=1}^n v_i = \hat{e}(K_1, P) \hat{e}(K_2, P) \cdots \hat{e}(K_n, P) = \hat{e}\left(\sum_{i=1}^n K_i, P\right) \\ f &= F_1(m) || (F_2(F_1(m)) \oplus m) \\ r &= H_1(v) + f \\ U_i &= K_i - rS_{ID_i} \end{aligned}$$

In the above computation, the symbol $||$ denotes concatenation of two operands.

Each signer transmits its individual signature (v_i, r, U_i) to the clerk who may be one of these signers or other specified trusted principal.

Once the clerk receives an individual signature (v_i, r, U_i) , he needs to verify the validity of this individual signature. The verification procedure of the clerk checks that

$$v_i = \hat{e}(U_i, P)\hat{e}(Q_{ID_i}, P_{pub})^r.$$

Once all individual signatures are received and verified by the clerk as valid, the multisignature of message m with respect to these signers who generate these individual signatures can be generated as (r, U) , where

$$U = \sum_{i=1}^n U_i = \sum_{i=1}^n K_i - r \sum_{i=1}^n S_{ID_i}$$

Verify: Given a multisignature (r, U) and n signer's identity $ID_1, ID_2, \dots, ID_n$ who stated have signed a message, a verifier computes

$$r - H_1(\hat{e}(U, P)\hat{e}(\sum_{i=1}^n Q_{ID_i}, P_{pub})^r) = f$$

and

$$m = [f]_{k_2} \oplus F_2([f]^{k_1}).$$

In the above computation, the subscript k_2 of f denotes the least significant k_2 bits of f , and the superscript k_1 of f denotes the most significant k_1 bits of f .

The verifier checks whether $[f]^{k_1} = F_1(m)$ holds. If this equation holds, the verifier accepts this multisignature and recovers the original message m from this multisignature. Otherwise, the verifier rejects the multisignature.

4.4 Security Analysis

Theorem 1. *This identity-based multisignature with message recovery scheme is correct and sound.*

Proof. The correctness of this identity-based multisignature with message recovery scheme can be shown as follows.

When the individual signature (v_i, r, U_i) is verified,

$$\begin{aligned} & \hat{e}(U_i, P)\hat{e}(Q_{ID_i}, P_{pub})^r \\ &= \hat{e}(K_i - rS_{ID_i}, P)\hat{e}(Q_{ID_i}, sP)^r \\ &= \hat{e}(K_i - rS_{ID_i}, P)\hat{e}(sQ_{ID_i}, P)^r \\ &= \hat{e}(K_i - rS_{ID_i}, P)\hat{e}(S_{ID_i}, P)^r \\ &= \hat{e}(K_i - rS_{ID_i}, P)\hat{e}(rS_{ID_i}, P) \\ &= \hat{e}(K_i, P) \\ &= v_i \end{aligned}$$

This means if the individual signature (v_i, r, U_i) is indeed generated by signer ID_i , the equation $v_i = \hat{e}(U_i, P)\hat{e}(Q_{ID_i}, P_{pub})^r$ will always hold.

When the multisignature (r, U) is verified, we can recover v which is used by each signer in the multisignature generation from the following computation.

$$\begin{aligned}
& \hat{e}(U, P)\hat{e}\left(\sum_{i=1}^n Q_{ID_i}, P_{pub}\right)^r \\
&= \hat{e}\left(\sum_{i=1}^n K_i - r \sum_{i=1}^n S_{ID_i}, P\right)\hat{e}\left(\sum_{i=1}^n Q_{ID_i}, sP\right)^r \\
&= \hat{e}\left(\sum_{i=1}^n K_i - r \sum_{i=1}^n S_{ID_i}, P\right)\hat{e}\left(s \sum_{i=1}^n Q_{ID_i}, P\right)^r \\
&= \hat{e}\left(\sum_{i=1}^n K_i - r \sum_{i=1}^n S_{ID_i}, P\right)\hat{e}\left(\sum_{i=1}^n S_{ID_i}, P\right)^r \\
&= \hat{e}\left(\sum_{i=1}^n K_i - r \sum_{i=1}^n S_{ID_i}, P\right)\hat{e}\left(r \sum_{i=1}^n S_{ID_i}, P\right) \\
&= \hat{e}\left(\sum_{i=1}^n K_i, P\right) \\
&= v
\end{aligned}$$

Then, using this v and part of the multisignature r , we can recover f from the following computation.

$$\begin{aligned}
& r - H_1(\hat{e}(U, P)\hat{e}\left(\sum_{i=1}^n Q_{ID_i}, P_{pub}\right)^r) \\
&= r - H_1(v) \\
&= H_1(v) + f - H_1(v) \\
&= f
\end{aligned}$$

Since f is computed from $f = F_1(m) || (F_2(F_1(m)) \oplus m)$, we will try to recover the original message m from f like this:

$$\begin{aligned}
& [f]_{k_2} \oplus F_2([f]^{k_1}) \\
&= [F_1(m) || (F_2(F_1(m)) \oplus m)]_{k_2} \oplus F_2([F_1(m) || (F_2(F_1(m)) \oplus m)]^{k_1}) \\
&= F_2(F_1(m)) \oplus m \oplus F_2(F_1(m)) \\
&= m
\end{aligned}$$

As previously declared, the subscript k_2 and the superscript k_1 of f denote the least significant k_2 and the most significant k_1 bits of f respectively.

After recovering the alleged original message m , we need to check whether $[f]^{k_1} = F_1(m)$ to verify the validity of the multisignature. If this equation holds, the multisignature (r, U) is valid and the original message m is recovered. Otherwise, the multisignature (r, U) is a forged one. $\square$

Theorem 2. *This identity-based multisignature with message recovery scheme is existentially unforgeable under a chosen message attack in the random oracle model, under the assumption that the Computational Diffie-Hellman problem is hard.*

Proof. Assume there is an algorithm $\mathcal{A}$ that can forge a multisignature under a chosen message attack. There will be another algorithm $\mathcal{B}$ that can run the algorithm $\mathcal{A}$ to solve the CDH problem.

In the process of $\mathcal{B}$ using $\mathcal{A}$ to solve the CDH problem, $\mathcal{B}$ needs to simulate all the oracles that $\mathcal{A}$ can query as follows.

Setup: $\mathcal{B}$ sets up $P_{pub} = aP$ as the system's public key and sends P_{pub} and other system parameters $\{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, q, P, H_1, H_2, F_1, F_2, k_1, k_2\}$ to adversary $\mathcal{A}$. In this case, $\mathcal{B}$ only knows the system's public key is aP , but he does not know the corresponding master-key s which is actually a in this concrete situation. Two hash functions F_1, F_2 of the four hash functions used in this scheme are published as normal hash functions. The other two hash functions H_1, H_2 are both treated as random oracles.

H_1 Queries: $\mathcal{B}$ creates and keeps two lists of tuples to simulate H_1 Oracle. At the beginning of the simulation, both of these lists are empty.

One list is called Hv_n -List which is used to store tuples like

$$(v_1, v_2, \dots, v_n, h).$$

In this type of tuples, the first n elements come from group $\mathbb{G}_2$ and the last element comes from $\mathbb{Z}_q^*$.

After receiving a H_1 hash query with respect to several elements $v_1, v_2, \dots, v_n$ in $\mathbb{G}_2$ and a message m , if the first n elements $v_1, v_2, \dots, v_n$ are not as a record in the v^* -List which is constructed in the Sign Oracle and not in a record in this Hv_n -List, $\mathcal{B}$ randomly selects $h \in \mathbb{Z}_q^*$ and returns h as the H_1 hash value of $v = \prod_{i=1}^n v_i$. Then, $\mathcal{B}$ records the tuple $(v_1, v_2, \dots, v_n, h)$ in this Hv_n -List. If the first n elements $v_1, v_2, \dots, v_n$ are already in a record in this Hv_n -List, $\mathcal{B}$ only returns the corresponding h in the record as the H_1 hash value. All in all, this list matches the situation that the honest signer is not required to participate in the multisignature generation.

The other list is called Hv^* -List which is used to store tuples like

$$(m, v_1, v_2, \dots, v_{n-1}, v^*, y - f).$$

In this type of tuples, the first element m is an arbitrary message to be signed by a subgroup which includes the honest signer. The next n elements come from group $\mathbb{G}_2$ and the last element comes from $\mathbb{Z}_q^*$.

After receiving an H_1 hash query with respect to several elements $v_1, v_2, \dots, v^*$ in $\mathbb{G}_2$ and a message m , if the first n elements $v_1, v_2, \dots, v_{n-1}, v^*$ are as a record in the v^* -List which is constructed in the Sign Oracle but not as a record in this Hv^* -List, $\mathcal{B}$ returns $y - f$ as the H_1 hash value of $v = \prod_{i=1}^{n-1} v_i \cdot v^*$ in which y is got from the corresponding record in the v^* -List and f is computed by the

equation $f = F_1(m) || (F_2(F_1(m)) \oplus m)$ with respect to the message m . Then, $\mathcal{B}$ records the tuple $(m, v_1, v_2, \dots, v_{n-1}, v^*, y-f)$ in this Hv^* -List. Note that for the same n elements $v_1, v_2, \dots, v_{n-1}, v^*$ but different message m , the value y is same because it comes from the same record in the v^* -List, but the value f is different because it is computed by the equation $f = F_1(m) || (F_2(F_1(m)) \oplus m)$ for different message. So, the returned hash value $y-f$ is different. In this case, we need to add a new record in this Hv^* -List. If these elements $m, v_1, v_2, \dots, v_{n-1}, v^*$ are already in a record in this Hv^* -List, $\mathcal{B}$ only returns the corresponding $y-f$ in the record as the H_1 hash value. In a word, this list matches the situation that the honest signer is required to participate in the multisignature generation.

H₂ Queries: $\mathcal{B}$ creates and keeps one list H_2 -List to simulate H_2 Oracle. At the beginning of the simulation, this list is empty.

For each H_2 hash query with respect to a signer ID_i except for the honest signer ID^* , if ID_i is not in a record in this H_2 -List, $\mathcal{B}$ randomly selects $k_i \in \mathbb{Z}_q^*$ and returns $Q_{ID_i} = k_i P$ as the H_2 hash value of ID_i . Then, $\mathcal{B}$ records the tuple (ID_i, k_i, Q_{ID_i}) in this H_2 -List. If ID_i is already in a record in this H_2 -List, $\mathcal{B}$ only returns the corresponding Q_{ID_i} in the record as the H_2 hash value.

For the H_2 hash query with respect to the honest signer ID^* , $\mathcal{B}$ returns $Q_{ID^*} = bP$ as the H_2 hash value of ID^* .

Extract Queries: $\mathcal{B}$ creates and keeps one list Ex-List to simulate Extract Oracle. At the beginning of the simulation, this list is empty.

For each Extract query with respect to a signer ID_i except for the honest signer ID^* , if ID_i is not in a record in this Ex-List, $\mathcal{B}$ looks up the H_2 -List which is created by H_2 Oracle to find the record about ID_i . Because a signer needs to query H_2 Oracle prior to its any other operation, the Extract Oracle can always find out the record with respect to ID_i in the H_2 -List. Using the k_i value in the record in the H_2 -List with respect to ID_i , $\mathcal{B}$ returns

$$S_{ID_i} = k_i P_{pub} = k_i aP = ak_i P = aQ_{ID_i}$$

as the signer ID_i 's private key. Then, $\mathcal{B}$ records the tuple (ID_i, S_{ID_i}) in this Ex-List. If ID_i is already in a record in this Ex-List, $\mathcal{B}$ only returns the corresponding S_{ID_i} in the record as the signer ID_i 's private key.

Sign Queries: $\mathcal{B}$ creates and keeps two lists of tuples to simulate Sign Oracle. At the beginning of the simulation, both of these lists v_n -List and v^* -List are empty. v_n -List matches the situation that the honest signer is not required to participate in the multisignature generation. v^* -List matches the situation that the honest signer is required to participate in the multisignature generation. Without loss of generality, we assume that the target signer is always the last signer ID_n .

For each Sign query with respect to an arbitrary message m and a subgroup of n signers $ID_1, ID_2, \dots, ID_n$, this oracle are divided into two phases.

In the first phase, $n-1$ signers $ID_1, ID_2, \dots, ID_{n-1}$ generate their individual $v_i = \hat{e}(K_i, P)$ in which K_i is randomly selected from group $\mathbb{G}_1$ and send their v_i and the target signer's identity ID_n to $\mathcal{B}$.

If ID_n is not the honest signer ID^* , $\mathcal{B}$ can randomly select an element K_n from group $\mathbb{G}_1$ and compute $v_n = \hat{e}(K_n, P)$. $\mathcal{B}$ returns v_n to $\mathcal{A}$ and records the tuple

$$(v_1, v_2, \dots, v_{n-1}, v_n, K_n)$$

in the v_n -List.

If ID_n is the honest signer ID^* , $\mathcal{B}$ can randomly select two integers $x, y \in_R \mathbb{Z}_q^*$. Then $\mathcal{B}$ computes

$$v^* = \hat{e}(aP, bP)^y \cdot \hat{e}(P, P)^x = \hat{e}((yab + x)P, P)$$

and returns this v^* to $\mathcal{A}$. In this case, the corresponding random element from group $\mathbb{G}_1$ is

$$K^* = (yab + x)P.$$

$\mathcal{B}$ records the tuple

$$(v_1, v_2, \dots, v_{n-1}, v^*, y, x)$$

in the v^* -List.

In the second phase, $\mathcal{A}$ computes $f = F_1(m) || (F_2(F_1(m)) \oplus m)$ with respect to message m . $\mathcal{A}$ queries H_1 Oracle the H_1 hash value with respect to $(v_1, v_2, \dots, v_{n-1}, v_n)$ or $(v_1, v_2, \dots, v_{n-1}, v^*)$ and message m and uses this H_1 hash value to compute the second part of $n - 1$ signer's individual signature (v_i, r, U_i) as $r = H_1(v) + f$. $\mathcal{A}$ computes the third part

$$U_i = K_i - rS_{ID_i} = K_i - raQ_{ID_i}$$

of $n - 1$ signer's individual signatures by the real Sign algorithm using the previous r and the corresponding private key $S_{ID_i} = aQ_{ID_i}$ got from the Extract Oracle and sends these $n - 1$ individual signatures and message m to $\mathcal{B}$.

$\mathcal{B}$ needs to compute $f = F_1(m) || (F_2(F_1(m)) \oplus m)$ at first. If ID_n is not the honest signer ID^* , $\mathcal{B}$ computes the individual signature (v_n, r, U_n) by the real Sign algorithm using the corresponding r which is computed the same as previous process and S_{ID_n} which is got from Extract Oracle. Then, $\mathcal{B}$ computes $U = \sum_{i=1}^n U_i$ and returns (r, U) as the multisignature on message m with respect to n signers $ID_1, ID_2, \dots, ID_n$. In this case, both of the individual signature (v_n, r, U_n) of ID_n and the multisignature (r, U) can pass their own verification process. These verifications can be checked by using the method in Theorem 4.1.

If ID_n is the honest signer ID^* , $\mathcal{B}$ computes r by using H_1 Oracle as

$$r = H_1(v) + f = y - f + f = y,$$

and simulates the third part of the honest signer ID^* 's individual signature as

$$U^* = K^* - rS_{ID^*} = (yab + x)P - y \cdot abP = xP$$

in which the corresponding x can be found out in the v^* -List.

Then, $\mathcal{B}$ computes

$$U = \sum_{i=1}^{n-1} U_i + U^*$$

and returns (r, U) as the multisignature on message m with respect to n signers $ID_1, ID_2, \dots, ID_{n-1}, ID^*$.

Verify: Both of the individual signature and the multisignature can pass the verifications. The individual signature (v^*, r, U^*) can pass the verification as follows.

$$\begin{aligned}
& \hat{e}(U^*, P) \hat{e}(Q_{ID^*}, P_{pub})^r \\
&= \hat{e}(xP, P) \hat{e}(bP, aP)^y \\
&= \hat{e}(xP, P) \hat{e}(yabP, P) \\
&= \hat{e}((yab + x)P, P) \\
&= v^*
\end{aligned}$$

The multisignature (r, U) can also pass the verification as follows.

$$\begin{aligned}
& \hat{e}(U, P) \hat{e}(\sum_{i=1}^n Q_{ID_i}, P_{pub})^r \\
&= \hat{e}(\sum_{i=1}^{n-1} U_i + U^*, P) \hat{e}(\sum_{i=1}^{n-1} Q_{ID_i} + Q_{ID^*}, aP)^y \\
&= \hat{e}(\sum_{i=1}^{n-1} U_i, P) \hat{e}(U^*, P) \hat{e}(\sum_{i=1}^{n-1} Q_{ID_i}, aP)^y \hat{e}(Q_{ID^*}, aP)^y \\
&= \hat{e}(\sum_{i=1}^{n-1} U_i, P) \hat{e}(\sum_{i=1}^{n-1} Q_{ID_i}, aP)^y \hat{e}(U^*, P) \hat{e}(Q_{ID^*}, aP)^y \\
&= \hat{e}(\sum_{i=1}^{n-1} K_i - ya \sum_{i=1}^{n-1} Q_{ID_i}, P) \hat{e}(ya \sum_{i=1}^{n-1} Q_{ID_i}, P) \hat{e}(xP, P) \hat{e}(yabP, P) \\
&= \hat{e}(\sum_{i=1}^{n-1} K_i, P) \hat{e}((yab + x)P, P) \\
&= \prod_{i=1}^{n-1} v_i \cdot v^*
\end{aligned}$$

Since we have assumed that adversary $\mathcal{A}$ can forge a multisignature under a chosen message attack, after the simulation process above, $\mathcal{A}$ can output a valid multisignature (r_1, U_1) on message m with respect to a subgroup of n signers which includes the honest signer ID^* who did not participate in the multisignature generation. There are two restrictions about this multisignature generation. The first one is there is no query to Extract Oracle with respect to the honest signer ID^* . The second one is there is no query to Sign Oracle with respect to the message m and a subgroup of signers which includes the honest signer ID^* . $\mathcal{B}$ can compute the third part U_1^* of the honest signer ID^* 's

individual signature (v_i^*, r_1^*, U_1^*) from the valid multisignature (r_1, U_1) as follows.

$$U_1^* = U - \sum_{i=1}^{n-1} U_i$$

All these U_i come from $\mathcal{A}$ in the second phase of the Sign Query.

$\mathcal{B}$ can reset all the oracles and runs $\mathcal{A}$ for the second time. At the end of the simulation, with a non-negligible probability $\mathcal{B}$ can get another different individual signature (v_2^*, r_2^*, U_2^*) on the same message m and with respect to the same honest signer ID^* when v_1^* equals to v_2^* . That means for two different random integer pairs (x_1, y_1) and (x_2, y_2) ,

$$K_1^* = (y_1 ab + x_1)P \text{ is equal to } K_2^* = (y_2 ab + x_2)P.$$

Both (r_1^*, U_1^*) and (r_2^*, U_2^*) can pass the verification process, and K_1^* equals K_2^* . So,

$$\begin{aligned} \hat{e}(U_1^*, P) \hat{e}(Q_{ID^*}, P_{pub})^{r_1^*} &= \hat{e}(U_2^*, P) \hat{e}(Q_{ID^*}, P_{pub})^{r_2^*} \\ \hat{e}(U_1^* + r_1^* S_{ID^*}, P) &= \hat{e}(U_2^* + r_2^* S_{ID^*}, P) \\ U_1^* + r_1^* S_{ID^*} &= U_2^* + r_2^* S_{ID^*} \\ (r_1^* - r_2^*) S_{ID^*} &= U_2^* - U_1^* \\ S_{ID^*} &= (r_1^* - r_2^*)^{-1} (U_2^* - U_1^*) \end{aligned}$$

In this case, $\mathcal{B}$ can compute the honest signer ID^* 's private key S_{ID^*} when he only knows the honest signer ID^* 's public key Q_{ID^*} and the system's public key P_{pub} . Because S_{ID^*} is expressed as abP , Q_{ID^*} is expressed as bP , P_{pub} is expressed as aP , $\mathcal{B}$ can solve an CDH problem if $\mathcal{A}$ is able to forge valid multisignatures.

If there is no such polynomial-time adversary that can forge a valid multisignature corresponding to a subgroup of signers that include an honest signer, we say that this identity-based multisignature with message recovery scheme is secure against existential forgery under chosen message attack. $\square$

5 Conclusion

We proposed a new notion of short identity-based multisignature scheme. The notion of short identity-based multisignature scheme can be viewed as identity-based multisignature with message recovery scheme. In order to sign short messages using a scheme that minimizes the total length of the original message and the appended signature, we proposed an concrete identity-based multisignature with message recovery scheme based on bilinear pairing in which multiple signers can generate a constant size multisignature on same message regardless the number of signers and there is no need to transmit the original message to verifier, because it can be recovered from the multisignature. We also proved that our scheme is secure against existential forgery on adaptively chosen message attack in the random oracle model, under the hardness assumption of CDH problem.

References

1. M. Abe and T. Okamoto. A signature scheme with message recovery as secure as discrete logarithm. *Advances in Cryptology-ASIACRYPT99*, pages 378–389, 1999.
2. A. Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. *Public Key CryptographyPKC 2003*, pages 31–46, 2002.
3. J. Cha and J. Cheon. An identity-based signature from gap diffie-hellman groups. In *Proceedings of the 6th International Workshop on Theory and Practice in Public Key Cryptography: Public Key Cryptography*, volume 2567, pages 18–30, 2003.
4. A. Fiat and A. Shamir. How to prove yourself: practical solutions to identification and signature problems. In *Advances in CryptologyCrypto86*, pages 186–194. Springer, 1987.
5. R. Gangishetti, M. Gorantla, M. Das, and A. Saxena. Identity based multisignatures. *Informatica*, 17(2):177–186, 2006.
6. L. Guillou and J. Quisquater. A paradoxical indentity-based signature scheme resulting from zero-knowledge. In *Advances in CryptologyCrypto88*, pages 216–231. Springer, 1990.
7. L. Harn and J. Ren. Efficient identity-based rsa multisignatures. *computers & security*, 27(1):12–15, 2008.
8. F. Hess. Efficient identity based signature schemes based on pairings. In *Selected Areas in Cryptography*, pages 310–324. Springer, 2003.
9. K. Itakura and K. Nakamura. A public-key cryptosystem suitable for digital multisignatures. *NEC Research & Development*, (71):1–8, 1983.
10. S. Micali, K. Ohta, and L. Reyzin. Accountable-subgroup multisignatures. In *Proceedings of the 8th ACM conference on Computer and Communications Security*, pages 245–254. ACM, 2001.
11. K. Nyberg and R. Rueppel. A new signature scheme based on the dsa giving message recovery. In *Proceedings of the 1st ACM conference on Computer and communications security*, pages 58–61. ACM, 1993.
12. T. Okamoto. Multi-signature schemes secure against active insider attacks. *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences*, 82(1):21–31, 1999.
13. A. Shamir. Identity-based cryptosystems and signature schemes. In *Advances in cryptology*, pages 47–53. Springer, 1985.
14. F. Zhang, W. Susilo, and Y. Mu. Identity-based partial message recovery signatures (or how to shorten id-based signatures). *Financial Cryptography and Data Security*, pages 578–578, 2005.