Journal of Information Hiding and Multimedia Signal Processing ©2016 ISSN 2073-4212
Ubiquitous International Volume **7**, Number **2**, March **2016**

An Improved Identity-Based Multi-Proxy Multi-Signature Scheme

Yan Xu, Hong Zhong* and Jie Cui

School of Computer Science and Technology
Anhui University
* *Corresponding author*
111 Jiulong Road, Hefei 230601, China
{xuyan,zhongh,cuijie}@ahu.edu.cn

Received April, 2015; revised November, 2015

Abstract. *In a multi-proxy multi-signature scheme, a group of original signers can delegate the signing rights to a group of proxy signers. All proxy signers cooperatively sign messages on behalf of the original group. Recently, Sahu and Padhye proposed an identity-based multi-proxy multi-signature (IBMPMS) scheme which was claimed to be secure against existential forgery on adaptive chosen-message and adaptive chosen-identity attacks in the random oracle model. However, in this paper, we indicate that Sahu-Padhye's scheme is insecure by giving concrete attacks. In the end, we propose a new IBMPMS scheme and prove that it is secure in the random oracle model.*
Keywords: Identity-based, Multi-proxy, Multi-signature, Random oracle model.

1. **Introduction.** In the traditional public key cryptography, certificate authority (CA) generates and signs digital certificates that bind a user's identity with its public key. It brings a heavy management burden for CA to operate and store the public key certificate. In order to resolve this problem, Shamir [1] introduced the concept of identity-based cryptography (IBC), where the public key is generated from user's unique identity such as phone number, email address and so on. A trusted third party named private key generator (PKG) generates private key using user's ID and PKG's master key.

Mambo *et al.* [2] introduced the concept of proxy signature in 1996. In a proxy signature scheme, an original signer can delegate its signing capability to a proxy signer to sign messages on its behalf. Proxy signature schemes have many practical applications such as mobile agent applications [3, 4], grid computing [5] and so on. Hwang and Chen [6] proposed a multi-proxy multi-signature scheme, where a group of original signers can delegate their signing rights to a group of proxy signers, and all proxy signers cooperatively sign messages on behalf of the original group. A number of multi-proxy multi-signature schemes with additional properties have been studied [7, 8, 9, 10] until now. Li and Chen [11] presented an IBMPMS scheme from bilinear pairings in 2005. Sahu and Padhye subsequently proposed several IBMPMS schemes [12, 13] without any formal security model. Recently, Sahu and Padhye [14] presented an IBMPMS scheme along with security model, and proved that this scheme is secure according to the security model. Unfortunately, we shall show that Sahu-Padhye's scheme is insecure in this paper. An adversary can forge a signature for any messages, or forge a delegation on any warrant w^* on behalf of any original signers. At the end, we propose a new IBMPMS scheme which is proven to be secure in the random oracle model under the computational Diffie-Hellman problem.

The rest of this paper is arranged as follows. In section 2, we give the fundamental knowledge of bilinear pairing, the definition and security model of Sahu-Radhye's IBMPMS scheme. We analysis the security of Sahu-Radhye's IBMPMS scheme in section 3. In section 4, we propose a new IBMPMS scheme and prove this scheme's security. Finally, section 5 gives a brief conclusion.

2. **Preliminaries.** In this paper, $[n]$ denotes $\{1, \ldots, n\}$, where n is a positive integer. We assume that there are n original signers A_i with identity ID_{A_i} respectively for $i \in [n]$, l proxy signers B_j with identity ID_{B_j} respectively for $j \in [l]$, where $n, l \in Z^+$.

2.1. **Bilinear Paring.** Let G_1 be a cyclic additive group and G_2 be a cyclic multiplicative group. G_1 and G_2 have the same prime order q. A bilinear map $e : G_1 \times G_1 \to G_2$ satisfies the following three properties:

(1) *Bilinearity:* $\forall a, b \in Z_q^*$ and $P, Q \in G_1$, we have $e(aP, bQ) = e(P, Q)^{ab}$.
(2) *Non-degeneracy:* There exist $P, Q \in G_1$, such that $e(P, Q) \neq 1$.
(3) *Computability:* $\forall P, Q \in G_1$, there exists an efficient algorithm to compute $e(P, Q) \in G_2$.

2.2. **Definition and Security Model for IBMPMS Scheme.** At present, the formal definition and security model of IBMPMS scheme [14] are recalled.

2.2.1. *Definition of IBMPMS scheme.* An IBMPMS scheme consists of the following algorithms:

Setup: On input a security parameter k, the PKG generates public parameters *params* and a master key *MSK*. Then, the PKG publishes *params* and keeps *MSK* confidential.

Extraction: On input the *MSK*, the public parameters *params* and a user's identity ID, the PKG outputs private key S_{ID} for the user.

Signature: On input a message m, the public parameters *params*, the signer's identity ID and private key S_{ID}, it outputs the message m's signature σ.

Verification: On input a signature σ, a message m, the public parameters *params*, the signer's ID, it returns 1 if σ is a valid signature on m for signer, otherwise returns 0.

Proxy key generation: On input a warrant w, all signers' identities ID_{A_i}, ID_{B_j}, private keys $S_{ID_{A_i}}, S_{ID_{B_j}}$ for $i \in [n], j \in [l]$, it outputs a partial proxy secret key S_{P_j} for each proxy signer B_j. w denotes the delegation warrant which includes the delegation police and the identities of all signers.

Multi-proxy multi-sign: On input the warrant w, a message m satisfying w, the partial proxy secret key S_{P_j}, for $j \in [l]$ of each proxy signer, it outputs an IBMPMS saying U_P on behalf of the original group.

Multi-proxy multi-sign verification: On input the public parameters *params*, the identities ID_{A_i}, ID_{B_j}, warrant w, message m and the IBMPMS U_P, it returns 1 if U_P is a valid multi-proxy multi-signature, otherwise returns 0.

2.2.2. *Security model.* Sahu and Radhye proposed the first formal security model for IBMPMS schemes in [14]. Here we recall Sahu-Radhye's security model according to the following game between a polynomial time adversary $\mathcal{A}$ and a challenger $\mathcal{B}$. f is a single honest user $\mathcal{A}$ tries to forge the IBMPMS scheme working against.

Setup: $\mathcal{B}$ runs the *Setup* algorithm to generate a master key *MSK* and public parameters *params*. $\mathcal{B}$ keeps *MSK* confidential and sends *params* to $\mathcal{A}$.

Extraction queries: $\mathcal{A}$ submits a user's identity ID (except for the user f). $\mathcal{B}$ runs the *Extraction* algorithm and returns the private key S_{ID} associated with ID to $\mathcal{A}$.

Signature queries: $\mathcal{A}$ chooses a message m and queries the standard signature for m. $\mathcal{B}$ responds a signature σ on behalf of the user f and adds the message m to list L_{S}.

Proxy key generation queries: There are two types of queries.

(i) $\mathcal{A}$ interacts with user B_f one of proxy signers. $\mathcal{A}$ submits a warrant w, and sends the delegation of all original signers on w to $\mathcal{B}$ that then runs the proxy key generation algorithm. Eventually, $\mathcal{B}$ returns a corresponding partial proxy signing key S_{P_f}, and adds the tuple $< w, S_{P_f} >$ to list L_{pk_f}. We stress that $\mathcal{A}$ has no right to access the element of L_{pk_p}.

(ii) $\mathcal{A}$ interacts with user A_f one of original signers. $\mathcal{A}$ submits a warrant w to $\mathcal{B}$. $\mathcal{B}$ responds proxy signing keys on w by running the proxy key generation algorithm, and adds the warrant w to list L_{pk_o}.

Multi-proxy multi-sign queries: $\mathcal{A}$ requests an IBMPMS on (m, w), where m satisfies w. If S_{P_f} exists such that $< w, S_{P_f} > \in L_{pk_f}$, $\mathcal{B}$ runs the multi-proxy multi-sign algorithm on (m, w) on behalf of f, outputs the partial proxy signature U_{P_f}, and forwards it to the proxy group's clerk, who combines all partial proxy signatures. Finally, $\mathcal{B}$ adds $< m, w >$ to list L_{mpms}.

Forgery: If any one event described as follows occurs, $\mathcal{A}$ wins the game.

- E_1: $\mathcal{A}$ forges a standard signature by user f for a message m that was not submitted to the **signature queries**.
- E_2: $\mathcal{A}$ forges an IBMPMS for a message m by proxy signers on behalf of original signers. Such that either the group of original signers never designated user B_f, or m was not submitted in the **multi-proxy multi-sign queries**.
- E_3: $\mathcal{A}$ forges an IBMPMS for a message m by proxy signers on behalf of original signers. Such that proxy signers were never designated by user A_f.

Definition 1. *An IBMPMS adversary $\mathcal{A}$ with $(t, q_H, q_E, q_S, q_{pk}, q_{mpms}, n + l, \epsilon)$ breaks the $n + l$ users IBMPMS scheme by the adaptive chosen-message and adaptive chosen-ID attacks, if $\mathcal{A}$ runs in at most t time; makes at most q_H hash queries; at most q_E extraction queries; at most q_S signature queries; at most q_{pk} proxy key generation queries; at most q_{mpms} multi-proxy multi-sign queries, and the success probability of $\mathcal{A}$ is at least ϵ.*

Definition 2. *An IBMPMS scheme is $(t, q_H, q_E, q_S, q_{pk}, q_{mpms}, n + l, \epsilon)-$secure against existential forgery on adaptive chosen-message and adaptive chosen-ID attacks, if no adversary $(t, q_H, q_E, q_S, q_{pk}, q_{mpms}, n + l, \epsilon)$ breaks it.*

3. Security analysis of Sahu-Radhye's IBMPMS scheme.

3.1. Review of Sahu-Radhye's scheme.

Sahu-Radhye's IBMPMS scheme[14] is recalled as follows.

Setup: For a security parameter k, the PKG chooses two cyclic group G_1 and G_2 with prime order q as well as a bilinear paring $e : G_1 \times G_1 \to G_2$, where G_1 is a cyclic additive group with generator P and G_2 is a cyclic multiplicative group. It also chooses a random value $s \in Z_q^*$, and three cryptographic hash functions $H_1 : \{0,1\}^* \to G_1$, $H_2 : \{0,1\}^* \times G_1 \to Z_q^*$, $H_3 : \{0,1\}^* \times G_2 \to Z_q^*$. Finally, the PKG sets $P_{pub} = sP$ and publishes the public parameters $params = \{q, G_1, G_2, e, P, P_{pub}, H_1, H_2, H_3\}$ while keeps the master key s secretly.

Extraction: The PKG generates public and private keys for all original and proxy signers. The public key of each original signer A_i is $Q_{ID_{A_i}} = H_1(\mathrm{ID}_{A_i})$, and the private key is $S_{ID_{A_i}} = sQ_{ID_{A_i}}$, respectively for $i \in [n]$. In the same way, the public and private keys of each proxy signer B_j ars $Q_{ID_{B_j}} = H_1(\mathrm{ID}_{B_j})$ and $S_{ID_{B_j}} = sQ_{ID_{B_j}}$, respectively for $j \in [l]$.

Signature: On input a message $m \in \{0,1\}^*$, the signer ID with private key S_{ID} randomly selects $v \in Z_q^*$, and computes $R = vP$, $h = H_2(m, R)$, $\sigma = hS_{ID}$. The signature on message m is (σ, R).
Verification: On input the signature (σ, R) on m, user ID's public key Q_{ID}, the verifier computes $h^* = H_2(m, R)$, and checks whether $e(P, \sigma) = e(P_{pub}, h^* Q_{ID})$. If the verification passes, (σ, R) is accepted as the signature.
Proxy key generation: Each proxy signer B_j, for $j \in [l]$ will get its proxy signing key S_{P_j} at the end of this phase.

(1) *Delegation:* To delegate its signing right, each original signer A_i, for $i \in [n]$ respectively signs on the warrant w which includes the delegation police, such as the identities of all signers, period of delegation and so on. Then, A_i performs the following jobs:

- Randomly chooses $v_i \in Z_q^*$, sets $V_i = v_i P$ and sends V_i to other original signers.
- Computes $V = \sum_{i=1}^{n} V_i$ and $S_{w_i} = H_2(w, V) S_{ID_{A_i}}$.
- Sends (S_{w_i}, w, V) to the group of proxy signers.

(2) *Verification of delegation:* Proxy signer B_j accepts (S_{w_i}, w, V), if $e(P, S_{w_i}) = e(P_{pub}, H_2(w, V) Q_{ID_{A_i}})$.

(3) *Proxy key generation:* After accepting the delegation, B_j computes the proxy signing key $S_{P_j} = S_w + H_2(w, V) S_{ID_{B_j}}$ where $S_w = \sum_{i=1}^{n} S_{w_i}$.
Multi-proxy multi-sign: In order to sign a message m satisfying the warrant w on behalf of the original group, each proxy signer B_j, for $j \in [l]$ operates as follows:

- Randomly chooses $x_j \in Z_q^*$, sets $r_{P_j} = e(P, P)^{x_j}$ and sends r_{P_j} to other proxy signers.
- Computes $r_P = \prod_{j=1}^{l} r_{P_j}$, $C_P = H_3(m, r_P)$ and $U_{P_j} = C_P S_{P_j} + x_j P$.
- Sends its partial proxy signature (C_P, U_{P_j}) to the clerk of the proxy group.

After receiving (C_P, U_{P_j}), the clerk validates the following equation:

$$r_{P_j} = e(U_{P_j}, P) e(\textstyle\sum_{i=1}^{n} Q_{ID_{A_i}} + Q_{ID_{B_j}}, H_2(w, V) P_{pub})^{-C_P}$$

If all equations hold, the clerk combines all proxy signatures (C_P, U_{P_j}) and generates the final IBMPMS (m, w, C_P, U_P, V) for the message m. Where $U_P = \sum_{j=1}^{l} U_{P_j}$.
Multi-proxy multi-sign verification: To verify the IBMPMS (m, w, C_P, U_P, V) on the message m under the warrant w, the verifier performs as follows:

- Computes $r_P = e(U_P, P) e(l \sum_{i=1}^{n} Q_{ID_{A_i}} + \sum_{j=1}^{l} Q_{ID_{B_j}}, H_2(w, V) P_{pub})^{-C_P}$.
- If $C_P = H_3(m, r_P)$, the verifier accepts the IBMPMS.

3.2. **Analysis of Sahu-Radhye IBMPMS scheme.** It is claimed that Sahu and Radhye's scheme [14] is secure in their security model. In this section, we will show that there is a polynomial time adversary $\mathcal{A}$ can always win the game between $\mathcal{A}$ and the challenger $\mathcal{B}$.

(1) In the **Setup** phase, the adversary $\mathcal{A}$ gets system's public parameters *params* from the challenger $\mathcal{B}$.
(2) In the **Extraction query** phase, $\mathcal{A}$ makes private key queries for any user ID (except for the user f which $\mathcal{A}$ wants to forge). $\mathcal{B}$ returns the private key S_{ID} to $\mathcal{A}$.
(3) In the **Signature query** phase, $\mathcal{B}$ returns the signature (σ, R) on a message m with respect to the user ID to $\mathcal{A}$, such that $e(P, \sigma) = e(P_{pub}, hQ_{ID})$.

3.2.1. *Forging signature for any message.* After receiving the signature (σ, R) on the message m, $\mathcal{A}$ calculates:

$$h = H_2(m, R) \quad \text{and} \quad S_{ID} = \sigma / h$$

$\mathcal{A}$ gets the private key S_{ID} of ID, and then can forge a signature on behalf of ID for any message m^* that was not submitted to the **signature query**. Furthermore, in the **Extraction query** and **Signature query** phase, $\mathcal{A}$ can get all signers' private keys, so it can forge IBMPMSs on any messages under any warrants.

3.2.2. *Forging warrant for any delegation.* $\mathcal{A}$ also can forge a valid delegation on any warrants w^* on behalf of an original signer A_f.

(1) After interacting with original signer A_f, $\mathcal{A}$ gets proxy signing key (S_w, w) of the original group on w from $\mathcal{B}$. $\mathcal{A}$ computes $h = H_2(w, V)$ and obtains the sum of all original signers' private keys $\sum_{i=1}^{n} S_{ID_{A_i}} = S_w / h$.

(2) $\mathcal{A}$ randomly selects $V^* \in G_1$ and forges a delegation on warrant w^*: $S_w^* = H_2(w^*, V^*) \sum_{i=1}^{n} S_{ID_{A_i}}$. $\mathcal{A}$ sends (S_w^*, w^*) to each proxy signer B_j. Then B_j generates its proxy signing key $S_{P_j}^* = S_w^* + H_2(w^*, V^*) S_{ID_{B_j}}, j \in [l]$.

(3) To sign a message m^* under the warrant w^* on behalf of the original group, B_j performs the following steps:

- Randomly chooses $x_j^* \in Z_q^*$, sets $r_{P_j}^* = e(P, P)^{x_j^*}$ and sends $r_{P_j}^*$ to other proxy signers.
- Computes $r_P^* = \prod_{j=1}^{l} r_{P_j}^*$, $C_P^* = H_3(m^*, r_P^*)$ and $U_{P_j}^* = C_P^* S_{P_j}^* + x_j^* P$.
- Sends its partial proxy signature $(C_P^*, U_{P_j}^*)$ to the clerk of the proxy group.

The clerk generates the final IBMPMS $(m^*, w^*, C_P^*, U_P^*, V^*)$ for the message m^* under the warrant w^*, where $U_P^* = \sum_{j=1}^{l} U_{P_j}^*$. It can be computed:

$$\begin{aligned} r_P^* &= \\ &= e(U_P^*, P) \\ &= e(l \sum_{i=1}^{n} Q_{ID_{A_i}} + \sum_{j=1}^{l} Q_{ID_{B_j}}, H_2(w^*, V^*) P_{pub})^{-C_P^*} \end{aligned}$$

and $C_P^* = H_3(m^*, r_P^*)$. Therefore, $(m^*, w^*, C_P^*, U_P^*, V^*)$ is a valid IBMPMS on the message m^* under the warrant w^*. It means that (S_w^*, w^*) is a valid delegation on w^* generated by the original group.

The section 3.2.1 and 3.2.2 imply that the events E_1, E_2, E_3 occur. That is, the probability that polynomial time adversary $\mathcal{A}$ wins the game is non-negligible. The attacks are successful for the reason that the basic signature presented by Sahu and Padhye is insecure and the generation of the proxy signing key is just a variant of the basic signature that signs on a warrant. Afterwards, we propose an improved scheme of Sahu-Padhye's IBMPMS scheme.

4. **The proposed IBMPMS scheme.** In Sahu-Padhye's IBMPMS scheme, the signature σ is a product of private key S_{ID} and a hash function H_1. It means the private key S_{ID} could be computed by σ / H_1. We will improve the scheme by introducing a secret value vP_{pub} in the signature σ.

4.1. **Scheme description.** ***Setup***, ***Extraction*** are same as described in section 3.
Signature: On input a message $m \in \{0,1\}^*$, the signer ID with private key S_{ID} randomly selects $v \in Z_q^*$, and computes $R = vP$, $h = H_2(m, R)$, $\sigma = hS_{ID} + vP_{pub}$. The signature on message m is (σ, R).
Verification: On input the signature (σ, R) on m, and ID's public key Q_{ID}, the verifier computes $h^* = H_2(m, R)$, and checks whether $e(P, \sigma) = e(P_{pub}, h^*Q_{ID} + R)$. If the verification passes, (σ, R) is accepted as the signature.
Proxy key generation: Each proxy signer $B_j, j \in [l]$ will get its proxy signing key S_{P_j} at the end of this phase.

(1) *Delegation:* To delegate the signing right, each original signer A_i, respectively for $i \in [n]$, signs on the warrant w. It performs the following jobs:

- Randomly chooses $v_i \in Z_q^*$, sets $V_i = v_iP$ and sends V_i to other original signers.
- Computes $V = \sum_{i=1}^{n} V_i$ and $S_{w_i} = H_2(w, V)S_{ID_{A_i}} + v_iP_{pub}$.
- Sends (S_{w_i}, w, V_i, V) to the proxy group.

(2) *Verification of delegation:* Proxy signer B_j accepts (S_{w_i}, w, V_i, V), if $e(P, S_{w_i}) = e(P_{pub}, H_2(w, V)Q_{ID_{A_i}} + V_i)$

(3) *Proxy key generation:* After accepting the delegation, B_j computes its proxy signing key $S_{P_j} = S_w + H_2(w, V)S_{ID_{B_j}}$, where $S_w = \sum_{i=1}^{n} S_{w_i}$.
Multi-proxy multi-sign: In order to sign a message m satisfying the warrant w on behalf of the original group, each proxy signer $B_j, j \in [l]$ operates as follows:

- Randomly chooses $x_j \in Z_q^*$, sets $r_{P_j} = e(P, P)^{x_j}$ and sends r_{P_j} to other proxy signers.
- Computes $r_P = \prod_{j=1}^{l} r_{P_j}$, $C_P = H_3(m, r_P)$ and $U_{P_j} = C_P S_{P_j} + x_jP$.
- Sends its partial proxy signature (C_P, U_{P_j}) to the clerk of the proxy group.

After receiving (C_P, U_{P_j}), the clerk validates the following equation:

$$\begin{aligned} & r_{P_j} \\ &= e(U_{P_j}, P)e(P_{pub}, V)^{-C_P} \\ &= e(\sum_{i=1}^{n} Q_{ID_{A_i}} + Q_{ID_{B_j}}, H_2(w, V)P_{pub})^{-C_P} \end{aligned}$$

If all equations hold, the clerk generates the final IBMPMS (m, w, C_P, U_P, V) for the message m. Where $U_P = \sum_{j=1}^{l} U_{P_j}$.
Multi-proxy multi-sign verification: To verify the IBMPMS (m, w, C_P, U_P, V) on a message m under the warrant w, the verifier performs as follows:

- Computes $r_P = e(U_P, P)e(H_2(w, V)\{l\sum_{i=1}^{n} Q_{ID_{A_i}} + \sum_{j=1}^{l} Q_{ID_{B_j}}\} + lV, P_{pub})^{-C_P}$.
- If $C_P = H_3(m, r_P)$, the verifier accepts the IBMPMS.

We can verify the correctness of the IBMPMS scheme as follows:

$$\begin{aligned} & e(U_P, P)e(H_2(w, V)\{l\textstyle\sum_{i=1}^{n} Q_{ID_{A_i}} + \sum_{j=1}^{l} Q_{ID_{B_j}}\} + lV, P_{pub})^{-C_P} \\ &= e(\textstyle\sum_{j=1}^{l} U_{P_j}, P)e(H_2(w, V)\{l\sum_{i=1}^{n} Q_{ID_{A_i}} + \sum_{j=1}^{l} Q_{ID_{B_j}}\} + lV, P_{pub})^{-C_P} \\ &= e(\textstyle\sum_{j=1}^{l} (C_P S_{P_j} + x_jP), P)e(H_2(w, V)\{l\sum_{i=1}^{n} Q_{ID_{A_i}} + \sum_{j=1}^{l} Q_{ID_{B_j}}\} + lV, P_{pub})^{-C_P} \\ &= e(\textstyle\sum_{j=1}^{l} x_jP, P) = r_P \end{aligned}$$

4.2. **Security analysis.** Based on the hardness assumption of the CDH problem, we can prove that our IBMPMS scheme is secure against existential forgery on adaptive chosen-message and adaptive chosen-ID attacks.

Theorem 1. In the random oracle model, if there exists an adversary $\mathcal{A}$ with $(t, q_H, q_E, q_S, q_{pk}, q_{mpms}, n+l, \epsilon)$ that can break our IBMPMS scheme, there exists an algorithm $\mathcal{B}$ that can solve the CDH problem with the probability ϵ' in time at most $t' \approx t + C_{G_1}(q_{H_1} + q_E + q_S + q_{pk} + q_{mpms} + 2)$.

Proof. Supposed that there exists an adversary $\mathcal{A}$ that can break the IBMPMS scheme with non-negligible probability, then we can construct a polynomial-time algorithm $\mathcal{B}$ to solve the CDH problem by using the adversary $\mathcal{A}$ as a subroutine.

Initialization. The algorithm $\mathcal{B}$ simulates the challenger and interacts with the adversary $\mathcal{A}$. $\mathcal{B}$ is given an instance $(P, sP, bP) \in G_1^3$ of CDH problem, and attemps to compute sbP.
Firstly, $\mathcal{B}$ sets $P_{pub} = sP$, and chooses three secure hash functions H_1, H_2, H_3. The system parameters is $params = \{G_1, G_2, e, P, P_{pub}, H_1, H_2, H_3\}$.
Queries. $\mathcal{A}$ can issue the following types of queries, $\mathcal{B}$ responses these queries and maintains lists $L_{H_1}, L_{H_2}, L_{H_3}$ that are initially empty for each hash query.

- H_1 queries: $\mathcal{A}$ issues a H_1 query for ID.
  1. If $\mathcal{B}$ finds a tuple (ID, h, a, c) in L_{H_1}, then it answers h as a response.
  2. Otherwise, $\mathcal{B}$ generates a random coin $c \in \{0, 1\}$ with probability $Pr[c = 0] = \mu$. $\mathcal{B}$ picks $a \in Z_q^*$ randomly, if $c = 0$, $\mathcal{B}$ sets $h = a(bP)$; else, $h = aP$, and adds (ID, h, a, c) in L_{H_1}.
- H_2 queries: $\mathcal{A}$ issues a H_2 query for (w, V).
  1. If $\mathcal{B}$ finds a tuple (w, V, t) in L_{H_2}, then it answers t as a response.
  2. Otherwise, $\mathcal{B}$ picks $t \in Z_q^*$ randomly, then it outputs t and adds (w, V, t) in L_{H_2}.
- H_3 queries:$\mathcal{A}$ issues a H_3 query for (m, r_p).
  1. If $\mathcal{B}$ finds a tuple (m, r_p, k_p) in L_{H_3}, then it answers k_p as a response.
  2. Otherwise, $\mathcal{B}$ picks $k_p \in Z_q^*$ randomly, then it outputs k_p and adds (m, r_p, k_p) in L_{H_3}.
- Extraction queries: $\mathcal{A}$ requests the private key corresponding to any signer ID, except for one original signer or one proxy signer. $\mathcal{B}$ runs H_1 **queries**, if $c = 1$, $\mathcal{B}$ outputs $S_{ID} = aP_{pub}$; else $\mathcal{B}$ aborts.
- Signature queries: $\mathcal{A}$ requests a standard signature on message $m \in \{0, 1\}$ of ID_f.
  1. $\mathcal{B}$ runs H_1 **queries** to obtain $Q_{ID_f} = a_{ID_f}P$.
  2. $\mathcal{B}$ picks $v \in Z_q^*$ randomly and sets $V = vP$. Then, $\mathcal{B}$ runs H_2 **queries** to obtain $H_2(m, V) = t$. Finally, $\mathcal{B}$ computes $\sigma = ta_{ID_f}P_{pub} + vP_{pub}$ and outputs (σ, R) as a valid signature on message m.

  The correctness of (σ, V) can be verified that:
  $e(\sigma, P) = e(ta_{ID_f}P_{pub} + vP_{pub}, P) = e(ta_{ID_f}P + V, P_{pub}) = e(tQ_{ID_f} + V, P_{pub})$.
- Proxy key generation queries: $\mathcal{A}$ requests such query on a warrant w for any proxy signer.
  1. $\mathcal{B}$ runs **Extraction queries** to obtain the private keys of proxy and original signers except for ID_f.
  2. $\mathcal{B}$ picks $v \in Z_q^*$ randomly and sets $V = vP$. Then, $\mathcal{B}$ runs H_2 **queries** to obtain $H_2(w, V) = t$.

3. $\mathcal{B}$ computes $S_w = \sum_{i=1}^{n} ta_{A_i} P_{pub} + vP_{pub}$ and generates the proxy key $S_{P_j} = S_w + ta_{B_j} P_{pub}$.

- Multi-proxy multi-sign queries: $\mathcal{A}$ requests such query on a message m, a warrant w.
  1. $\mathcal{B}$ runs **Extraction queries**, **Proxy key generation queries** to obtain private keys and proxy keys except for ID_f.
  2. $\mathcal{B}$ randomly picks $x \in Z_q^*$, sets $X = xP, r_p = e(P,P)^x$, and then runs H_3 **queries** to obtain $H_3(m, r_p) = k_p$.
  3. $\mathcal{B}$ computes $U_p = X + k_p t\{l(\sum_{i=1}^{n} a_{A_i} P_{pub}) + \sum_{j=1}^{l} a_{B_j} P_{pub}\}$.

We can check the validity of provided IBMPMS scheme as follows:
$e(U_P, P)e(H_2(w,V)\{l\sum_{i=1}^{n} Q_{ID_{A_i}} + \sum_{j=1}^{l} Q_{ID_{B_j}}\} + lV, P_{pub})^{-C_P} = r_p$

Forgery. The adversary $\mathcal{A}$ outputs a valid IBMPMS $(m^*, w^*, C_p^*, U_p^*, V^*)$ for the message m^*, the warrant w^*. We will show that algorithm $\mathcal{B}$ can solve an instance of the CDH problem according two cases.

Case 1. f is a original signer. In this case, $\mathcal{A}$ did not request the private key of ID_{A_f}, the proxy key generated by ID_{A_f} for w^*, the IBMPMS of (ID_{A_f}, m^*, w^*). Without loss of generality, we assume that $ID_{A_f} = ID_{A_1}$. Firstly, $\mathcal{B}$ runs H_1 **queries** for ID_{A_1}, if $c_1 = 1$, then $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ computes $U_p' = U_P^* - [k_p t\{l\sum_{i=2}^{n} a_{A_i} + \sum_{j=1}^{j} a_{B_j}\}P_{pub} + X]$. Then, $\mathcal{B}$ gets $U_p' = (k_p t l a_{A_1}) sbP$, and then obtains $sbP = (k_p t l a_{A_1})^{-1} U_p'$.

We can easily see that $\mathcal{B}$ is a polynomial time algorithm and the probability for $\mathcal{B}$ to solve the CDH problem is $\epsilon' = \epsilon\mu(1-\mu)^{q_E+q_S+nq_{pk}+(n+l-1)(q_{mpms}+1)}$ that is non-negligible.

Case 2. f is a proxy signer. In this case, $\mathcal{A}$ did not request the private key of ID_{B_f}, the proxy key of (ID_{B_f}, w^*), the IBMPMS of (ID_{B_f}, m^*, w^*). Without loss of generality, we assume that $ID_{B_f} = ID_{B_1}$. Firstly, $\mathcal{B}$ runs H_1 **queries** for ID_{B_1}, if $c_1 = 1$, then $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ computes $U_p' = U_P^* - [k_p t\{l\sum_{i=1}^{n} a_{A_i} + \sum_{j=2}^{j} a_{B_j}\}P_{pub} + X]$. Then, $\mathcal{B}$ gets $U_p' = (k_p t a_{B_1}) sbP$, and then obtains $sbP = (k_p t a_{B_1})^{-1} U_p'$. In this case, the probability ϵ' for $\mathcal{B}$ to solve the CDH problem is non-negligible too with $\epsilon' = \epsilon\mu(1-\mu)^{q_E+q_S+nq_{pk}+(n+l-1)(q_{mpms}+1)}$.

Therefor, the success probability that $\mathcal{B}$ solves the above instance of CDH problem is non-negligible. This completes the proof.

5. **Conclusions.** Sahu and Padhye proposed an identity-based multi-proxy multi-signature scheme from bilinear pairings. They claimed that this scheme was secure under their secure model. However, we have proved that Sahu-Padhye's IBMPMS scheme is insecure by concrete attacks. We also give an improved scheme to prevent the attacks. The new scheme is secure under the computational Diffie-Hellman problem in random oracle model.

Acknowledgment. This work is partially supported by National Natural Science Foundation of China(61173188, 61572001), the Open Project of Co-Innovation Center for Information Supply & Assurance TechnologyAnhui University (ADXXBZ2014-9), China Postdoctoral Science Foundation(2015M570545), and Anhui Provincial Natural Science Foundation (201508085QF132).

REFERENCES

[1] A. Shamir, Indentity based cryptosystem and signature scheme, *Proc. of Crypto,* Berlin Heidelberg, Springer-Verlag, vol.196, pp.47-53, 1984.

[2] M. Mambo, K. Usuda, E. Okamoto, Proxy signatures: Delegation of the power to sign messages, *IEICE transactions on fundamentals,* vol. E79-A, no. 9, pp.1338-1354, 1996.
[3] B. Lee, H. Kim, K. Kim, Strong proxy signature and its applications, *Proc. of SCIS*, vol.1, pp.603-608, 2001.
[4] H. Kim, J. Baek, B. Lee, K. Kim, Secret computation with secrets for mobile agent using one-time proxy signature, *Proc. of SCIS*, vol. 1, pp.845-850, 2001.
[5] I. Foster, C. Kesselman, G. Tsudik, S. Tuecke, A security architecture for computational grids, *Proc. of the 5th ACM Conference on CCS,* ACM, NewYork, USA, pp.83-92, 1998.
[6] S. Hwang, C. Chen, New multi-proxy multi-signature schemes, *Applied Mathematics and Computation,* vol. 147, no. 1, pp.57-67, 2004.
[7] S. F. Tzeng, C. Y. Yang, M. S. Hwang, A nonrepudia-ble threshold multi-proxy multi-signature scheme with shared verification, *Future Generation Computer Sys-tems,* vol. 20, no. 5, pp.887-893, 2004.
[8] T. S. Wu, H. Y. Lin, P. Y. Ting, A publicly verifiable PCAE scheme for confidential applications with proxy dele-gation, *Transactions on Emerging Telecommunications Technologies,* vol. 23, no. 2, pp.172-185, 2012.
[9] S. Padhye, N. Tiwari, ECDLP-based certificateless proxy signature scheme with message recovery, *Transactions on Emerging Telecommunications Technologies,* vol.26, no.3, pp.346-354, 2015.
[10] M. R. Asaar, M. Salmasizadeh, W. Susilo, An identity-based multi-proxy multi-signature scheme without bilinear pairings and its variants, *The Computer Journal* vol. 58, no. 4, pp.1021-1039, 2015.
[11] X. Li, K. Chen, ID-based multi-proxy signature, proxy multi-signature and multi-proxy multi-signature schemes from bilinear pairings, *Applied Mathematics and Computation,* vol.169, no.1, pp.437-450, 2005.
[12] R. A. Sahu, S. Padhye, An ID-based multi-proxy multi-signature scheme, *Proc. of IEEE International Conference on Computer and Communication Technology*, pp. 60-63, 2010.
[13] R. A. Sahu, S. Padhye, ID-based multi-proxy multi-signature scheme from bilinear pairing, *Proc. of 5th WSEAS International Conference on Computer Engineering and Applications*, IEEE, Allahabad, Uttar Pradesh, pp.43-48, 2011.
[14] R. A. Sahu, S. Padhye, Identity-based multi-proxy multi-signature scheme provably secure in random oracle model, *Transactions on Emerging Telecommunications Technologies,* vol. 26, no. 4, pp.547-558, 2015.