On the Security of an Improved Identity-based Proxy Signature Scheme without Random Oracles

Caixue Zhou, Zongmin Cui, Guangyong Gao

School of Information Science and Technology, Jiujiang University, China
charlesjjjx@126.com, cuizm01@gmail.com, gaoguangyong@163.com

Abstract

Proxy signature can realize that an original signer delegates his/her signing right to a proxy signer. Then, the proxy signer can sign messages on behalf of the original signer when he/she is absent. The identity-based cryptosystem can simplify the costly certificate management. In this paper, we demonstrate that an improved identity-based proxy signature scheme in the standard model is not secure by giving four kinds of attacks. An improved scheme is also proposed to overcome the security flaws. Our improved scheme can be proved secure assuming the CDH problem to be hard. Performance analysis shows that our improved scheme is practical.

Keywords: Identity-based proxy signature, Proxy signature, Bilinear pairing, Standard model, Computational Diffie-Hellman assumption

1 Introduction

The identity based cryptosystem can simplify the costly certificate management which is considered to be the main drawback of the traditional public key cryptosystem. This cryptographic concept was first introduced by Shamir [1] in 1984. But an efficient identity-based encryption scheme was not invented until Boneh and Franklin [2] proposed it by using bilinear pairings in 2001. Since then, the identity based cryptosystem has become a research hotspot.

Proxy signature is a useful tool when an original signer is absent. Then he/she can delegate his/her signing right to a proxy signer. Any verifier can be convinced that the signature is made by the proxy signer designated by the original signer. This cryptographic primitive was first introduced by Mambo et al. [3] in 1996. Proxy signature has also got a lot of attention since it was introduced.

Combining the identity based cryptosystem and proxy signature, Zhang and Kim [4] first introduced the identity based proxy signature by using bilinear pairings in 2003, but their scheme lacked security proof. In 2005, Xu et al. [5] gave a formal definition and security model for identity-based proxy signature for the first time. Their security model was based on Boldyreva et al.'s work [6]. In 2006, Huang et al. [7] proposed a proxy signature scheme in the standard model for the first time. In the same year, Galindo et al. [8] gave a generic construction of identity based proxy signature from traditional public key based proxy signature and their construction suits in the standard model. In 2010, Cao and Cao [9] proposed a direct construction of identity based proxy signature in the standard model for the first time. But in 2013, Sun et al. [10] pointed out that Cao et al.'s scheme suffers from a malicious original signer attack and a malicious proxy signer attack. In the same year, Gu et al. [11] proposed another identity-based proxy signature scheme in the standard model. Unfortunately, He et al. [12] gave out three kinds of attacks to Gu et al.'s scheme and Hu et al. [13] gave out four kinds of attacks to Gu et al.'s scheme in 2015, respectively. Based on Gentry's identity based encryption scheme [14], Hu et al. [15] proposed another highly efficient identity-based proxy signature scheme in the standard model in 2014. Based on Tian et al.'s strong designated verifier signature scheme [16], Hu et al. [17] also proposed an identity-based proxy signature scheme in the standard model with tight security reduction in 2015.

To overcome the security flaws of Gu et al.'s scheme [11], Hu et al. [18] proposed an improved scheme in 2017. They gave a security proof to their scheme. But unfortunately, in this paper, we point out that Hu et al.'s improved scheme is still insecure. We give four kinds of attacks to their scheme. Then we give further improvement to their scheme. We give security proof and efficiency analysis of our scheme. The performance evaluation shows that our scheme is practical.

The rest of the paper is organized as follows. In Section 2, we introduce the concept of bilinear pairing, the complexity assumption, the formal definition and security model of identity-based proxy signature. In Section 3, we give a description of Hu et al.'s scheme. In Section 4, we give four kinds of attacks to Hu et al.'s scheme. In Section 5, we propose an improved scheme. In Section 6, we discuss the correctness,

*Corresponding Author: Caixue Zhou; E-mail: charlesjjjx@126.com
DOI: 10.3966/160792642018121907009

security and efficiency of our improved scheme. In Section 7, we give an application example of our scheme. We conclude the paper in Section 8.

2 Preliminaries

2.1 Bilinear Pairing

Let G_1, G_2 be two multiplicative cyclic groups of prime order q and g be a generator of G_1. The map $e: G_1 \times G_1 \to G_2$ is said to be an admissible bilinear pairing if the following three conditions hold.

(1) Bilinearity: For all $a, b \in Z_q$, $P, Q \in G_1$, we have $e(P^a, Q^b) = e(P, Q)^{ab}$.

(2) Non-degeneracy: $e(g, g) \neq 1_{G_2}$.

(3) Computability: For all $P, Q \in G_1$, there exists an efficient algorithm to compute $e(P, Q)$.

2.2 Complexity Assumption

Computational Diffie-Hellman (CDH) problem. Given $g, g^a, g^b \in G_1$ for unknown randomly chosen $a, b \in Z_q$, one must compute g^{ab}.

The $(\varepsilon, t)-$ CDH assumption. No probabilistic polynomial time (PPT) algorithm A running in a maximum time of t with a probability of at least ε can solve the CDH problem in G_1.

2.3 Formal Definition

The formal definition is the same as that in Hu et al.'s scheme [18]. An identity-based proxy signature scheme consists of the following eight algorithms.

Setup. Given a security parameter 1^k, the PKG produces a master private key s and the system public parameters $Params$. $Params$ are public to all while s is kept private by the PKG.

KeyGen. Given an identity ID and $Params$, the PKG uses the master private key s to produce ID's private key SK_{ID}. Then the PKG sends SK_{ID} to the user secretly. Thus, the original signer's identity and private key pair is (ID_a, SK_{ID_a}) and the proxy signer's identity and private key pair is (ID_p, SK_{ID_p}).

ISign. Given a private key SK_{ID} of identity ID, a message m and $Params$, the signer ID produces a standard signature σ.

IVerify. Given the signer's identity ID, the signature σ, the message m and $Params$, the verifier verifies the standard signature σ and outputs true or false.

IDelegate. Given the private key SK_{ID_a} of an original signer ID_a, $Params$ and a warrant m_w (which includes the identities of the original signer and proxy signer, the types of delegated message, the delegation period and so on), the original signer ID_a produces a delegation δ. Then he/she sends it to the proxy signer.

IProxyKeyGen. Given the private key SK_{ID_p} of the proxy signer, the warrant m_w, the delegation δ and $Params$, the proxy signer ID_p produces a proxy signing key PSK_{ID_p}.

IProxySign. Given the proxy signing key PSK_{ID_p}, a message m, a warrant m_w and $Params$, the proxy signer ID_p produces a proxy signature σ_p.

IProxyVerify. Given the identities of the original signer ID_a and proxy signer ID_p, the warrant m_w, the message m, the proxy signature σ_p and $Params$, the verifier verifies the proxy signature σ_p and outputs true or false.

For consistency, we require if $\sigma = ISign(SK_{ID}, m, Params)$, then $IVerify(\sigma, ID, m, Params) = true$. And if $\sigma_p = I\,\mathrm{Pr}oxySign(PSK_{ID_p}, m, m_w, Params)$, then $I\,\mathrm{Pr}oxyVerify(\sigma_p, ID_a, ID_p, m, m_w, Params) = true$.

2.4 Security Model

Based on the security models of Boldyreva et al. [6] and Schuldt et al. [19], Gu et al. [11] introduced a more complete security model of identity-based proxy signature. Hu et al. [18] used the same security model as Gu et al. In the security model, they classified the proxy signature to several types. Here we must point out that there is a general classification of proxy signature in Liu et al.'s scheme [20]. Their security model is as follows.

It assumes that only one user u^* is not corrupted, that is, the adversary A can get all useful information except the private key of u^*. There are four situations to be considered.

(1) The adversary A forges a standard signature of u^*.

(2) The adversary A does not get the proxy signing key of u^* and forges a proxy signature of u^*, where u^* is both the original signer and the proxy signer.

(3) The adversary A does not get the proxy signing key of u^* and forges a proxy signature of u^*, where u_i ($u_i \neq u^*$) is the original signer and u^* is the proxy signer.

(4) The adversary A does not get the signing rights of u^* and proxy signing key of u_i ($u_i \neq u^*$). He/she forges a proxy signature of u_i ($u_i \neq u^*$), where u^* is the original signer and u_i ($u_i \neq u^*$) is the proxy signer.

For simplicity, it assumes that user ID_1 is the non-corrupted user. The security model is described as

follows.

Setup: The challenger C runs the setup algorithm to produce the public system parameters *Params* and a master private key s. C gives *Params* to A while keeping s private.

Queries. A can make a polynomially bounded number of queries as follows.

Key queries.

(1) T=Type1 (key-oracle()): A requests the private key of a user ID_i $(i>1)$, C produces a private key SK_{ID_i} and returns it to A.

(2) T=Type2 (proxykey-oracle1()): A supplies an identity ID_i $(i\geq 1)$ and a warrant m_w, where ID_i $(i\geq 1)$ is both the original signer and the proxy signer. C produces a self-delegation proxy key PSK_{ID_i} and returns it to A.

(3) T=Type3 (proxykey-oracle2()): A supplies an identity ID_i $(i>1)$ and a warrant m_w, where ID_i $(i>1)$ is the original signer and ID_1 the proxy signer. C produces a proxy key PSK_{ID_1} and returns it to A.

(4) T=Type4 (proxykey-oracle3()): A supplies an identity ID_i $(i>1)$ and a warrant m_w, where ID_1 is the original signer and ID_i $(i>1)$ the proxy signer. C produces a proxy key PSK_{ID_i} and returns it to A.

Signature queries.

(1) T=Type1 (sign-oracle()): A supplies an identity ID_i $(i\geq 1)$ and a message m. C produces a standard signature σ of ID_i $(i\geq 1)$ and returns it to A.

(2) T=Type2 (psign-oracle1()): A supplies an identity ID_i $(i\geq 1)$, a warrant m_w and a message m, where ID_i $(i\geq 1)$ is both the original signer and the proxy signer. C produces a proxy signature σ_p and returns it to A.

(3) T=Type3 (psign-oracle2()): A supplies an identity ID_i $(i>1)$, a warrant m_w and a message m, where ID_i $(i>1)$ is the original signer and ID_1 the proxy signer. C produces a proxy signature σ_p and returns it to A.

(4) T=Type4 (psign-oracle3()): A supplies an identity ID_i $(i>1)$, a warrant m_w and a message m, where ID_1 is the original signer and ID_i $(i>1)$ the proxy signer. C produces a proxy signature σ_p and returns it to A.

Forgery.

(1) T=Type1: A outputs a forged standard signature σ^* on (ID^*, m^*). If the following conditions hold, then we say that A wins the game.

(a) $IVerify(Params, \sigma^*, m^*, ID^*) = true$;

(b) A did not make key-oracle() on ID^*;

(c) A did not make sign-oracle() on (ID^*, m^*).

(2) T=Type2: A outputs a forged self-delegation proxy signature σ_p^* on (m_w^*, ID^*, m^*), where ID^* is both the original signer and the proxy signer. If the following conditions hold, then we say that A wins the game.

(a) $I\Pr oxyVerify(Params, \sigma_p^*, m_w^*, m^*, ID^*) = true$;

(b) A did not make key-oracle() on ID^*;

(c) A did not make proxykey-oracle1() on (ID^*, m_w^*);

(d) A did not make psign-oracle1() on (m_w^*, ID^*, m^*).

(3) T=Type3: A outputs a forged proxy signature σ_p^* on (m_w^*, ID^*, ID_1, m^*), where ID^* is the original signer and ID_1 the proxy signer. If the following conditions hold, then we say that A wins the game.

(a) $I\Pr oxyVerify(Params, \sigma_p^*, m_w^*, m^*, ID^*, ID_1) = true$;

(b) A did not make proxykey-oracle2() on (ID^*, ID_1, m_w^*);

(c) A did not make psign-oracle2() on (m_w^*, ID^*, ID_1, m^*).

(4) T=Type4: A outputs a forged proxy signature σ_p^* on (m_w^*, ID^*, ID_1, m^*), where ID_1 is the original signer and ID^* the proxy signer. If the following conditions hold, then we say that A wins the game.

(a) $I\Pr oxyVerify(Params, \sigma_p^*, m_w^*, m^*, ID^*, ID_1) = true$;

(b) A did not make proxykey-oracle3() on (ID^*, ID_1, m_w^*);

(c) A did not make psign-oracle3() on (m_w^*, ID^*, ID_1, m^*).

We say that an adversary A can $(t, \varepsilon, q_e, q_s)$ break an identity-based proxy signature scheme if A makes at most q_e key queries and q_s signature queries, and runs in a maximum time t with a probability of at least ε.

An identity-based proxy signature scheme is $(t, \varepsilon, q_e, q_s)$ secure if no PPT adversary can $(t, \varepsilon, q_e, q_s)$ break it.

3 Review of Hu et al.'s Scheme

Setup: Given a security parameter 1^k, the PKG chooses two cyclic multiplicative groups G_1 and G_2 of prime order q, a random generator g of G_1, a bilinear map: $e: G_1 \times G_1 \to G_2$ and a hash function $H: \{0,1\}^* \to Z_q$. The PKG randomly chooses $\alpha \in Z_q$ and $g_2, \mu_0, \mu, v_0, v, \varpi, \tau \in G_1$. The PKG sets $g_1 = g^\alpha$. The system public parameters are $Params = \{G_1, G_2, e, g, g_1, g_2, \mu_0, \mu, v_0, v, \varpi, \tau, H\}$. The system master private key is α.

KeyGen. Given an identity ID, the PKG randomly

chooses $r_{ID} \in Z_q$ and computes $x_{ID,0} = g_2^{\alpha} \cdot (\mu \cdot \mu_0^{H(ID)})^{\alpha \cdot r_{ID}}$ and $x_{ID,1} = g_1^{r_{ID}}$. The private key of user ID is $sk_{ID} = (x_{ID,0}, x_{ID,1})$. Then the original signer ID_a's private key is $sk_{ID_a} = (x_{ID_a,0}, x_{ID_a,1}) = (g_2^{\alpha} \cdot (\mu \cdot \mu_0^{H(ID_a)})^{\alpha \cdot r_{ID_a}}, g_1^{r_{ID_a}})$ and the proxy signer ID_p's private key is $sk_{ID_p} = (x_{ID_p,0}, x_{ID_p,1}) = (g_2^{\alpha} \cdot (\mu \cdot \mu_0^{H(ID_p)})^{\alpha \cdot r_{ID_p}}, g_1^{r_{ID_p}})$.

ISign. Given an identity ID and a message m, the signer randomly chooses $d \in Z_q$ and computes $X_0 = x_{ID,0} \cdot \varpi^d \cdot \tau^{d \cdot H(m)}$, $X_1 = x_{ID,1} = g_1^{r_{ID}}$, $X_2 = g^d$. Finally, the signature is $\sigma = (m, X_0, X_1, X_2)$.

Iverify. The verifier checks whether the following equation holds. $e(X_0, g) = e(g_2, g_1) \cdot e(\mu \cdot \mu_0^{H(ID)}, X_1) \cdot e(\varpi, X_2) \cdot e(\tau, X_2^{H(m)})$.

IDelegate. The original signer ID_a produces a warrant m_w, which contains the descriptions of delegation duration, delegation message type, the identities of original signer and proxy signer and so on. Then he/she randomly chooses $s \in Z_q$ and computes $T_0 = x_{ID_a,0} \cdot (v \cdot v_0^{H(m_w)})^s$, $T_1 = x_{ID_a,1} = g_1^{r_{ID_a}}$, $T_2 = g^s$. Finally, the delegation is $\delta = (m_w, T_0, T_1, T_2)$.

IProxyKeyGen. The proxy signer ID_p first checks whether the delegation is valid by checking the following equation. $e(T_0, g) = e(g_2, g_1) \cdot e(\mu \cdot \mu_0^{H(ID_a)}, T_1) \cdot e(v \cdot v_0^{H(m_w)}, T_2)$. If it is not true, then he/she requests the original signer to reproduce the delegation $\delta = (m_w, T_0, T_1, T_2)$. Otherwise, he/she computes $y_0 = x_{ID_p,0} \cdot T_0$, $y_1 = x_{ID_p,1} = g_1^{r_{ID_p}}$, $y_2 = T_1 = g_1^{r_{ID_a}}$, $y_3 = T_2 = g^s$. Finally, the proxy key is $PSK_{ID_p} = (y_0, y_1, y_2, y_3)$.

IProxySign. Given a message m, the proxy signer ID_p randomly chooses $d \in Z_q$ and computes $Y_0 = y_0 \cdot \varpi^d \cdot \tau^{d \cdot H(m)}$, $Y_1 = y_1 = g_1^{r_{ID_p}}$, $Y_2 = y_2 = g_1^{r_{ID_a}}$, $Y_3 = y_3 = g^s$, $Y_4 = g^d$. Finally, the proxy signature is $\sigma_p = (m, m_w, Y_0, Y_1, Y_2, Y_3, Y_4)$.

IProxyVerify. The verifier checks whether the following equation holds. $e(Y_0, g) = e(g_2, g_1)^2 \cdot e(\mu \cdot \mu_0^{H(ID_p)}, Y_1) \cdot e(\mu \cdot \mu_0^{H(ID_a)}, Y_2) \cdot e(v \cdot v_0^{H(m_w)}, Y_3) \cdot e(\varpi, Y_4) \cdot e(\tau, Y_4^{H(m)})$.

4 Analysis of Hu et al.'s Scheme [18]

Hu et al. [18] pointed out that Gu et al.'s scheme [11] is insecure by demonstrating a concrete attack. In order to overcome the security flaw, Hu et al. introduced two public system parameters μ_0 and v_0. The KeyGen algorithm becomes $x_{ID,0} = g_2^{\alpha} \cdot (\mu \cdot \mu_0^{H(ID)})^{\alpha \cdot r_{ID}}$ and $x_{ID,1} = g_1^{r_{ID}}$ instead of $x_{ID,0} = g_2^{\alpha} \cdot \mu^{H(ID) \cdot \alpha \cdot r_{ID}}$ and $x_{ID,1} = g_1^{r_{ID}}$, and the IDelegate algorithm becomes $T_0 = x_{ID_a,0} \cdot (v \cdot v_0^{H(m_w)})^s$, $T_1 = x_{ID_a,1} = g_1^{r_{ID_a}}$ and $T_2 = g^s$ instead of $T_0 = x_{ID_a,0} \cdot v^{H(m_w) \cdot s}$, $T_1 = x_{ID_a,1} = g_1^{r_{ID_a}}$ and $T_2 = g^s$. After making these improvements, the scheme can resist Hu et al.'s attack [18]. But unfortunately, we find that the scheme is still insecure. In the following, we will show four kinds of attacks to Hu et al.'s scheme [18].

4.1 Forging a Proxy Signature 1

In the following, we will show that after getting a valid $T = type3$ proxy signature, the adversary A can change the original signer to another person and forge another $T = type3$ proxy signature.

According to Hu et al.'s security model, by maximizing the adversary's attack abilities, it assumes that just one user u^* is not corrupted. Let's consider ID_{a1} and ID_{a2} who are corrupted by the adversary A, that is, the private key $(x_{ID_{a1},0}, x_{ID_{a1},1})$ of ID_{a1} and $(x_{ID_{a2},0}, x_{ID_{a2},1})$ of ID_{a2} are known by A.

(1) A sets ID_{a1} as the original signer and u^* as the proxy signer. A produces a warrant m_w. A produces a delegation $(T_{a1,0}, T_{a1,1}, T_{a1,2})$ on (ID_{a1}, u^*, m_w).

(2) In the signature queries stage, A makes a $T = type3$ (psign-oracle2()) oracle query, where ID_{a1} is the original signer, u^* is the proxy signer, m_w is the warrant and m is the message. The challenger C returns a proxy signature $\sigma_p = \{Y_0, Y_1, Y_2, Y_3, Y_4\}$ to A.

(3) Forgery: After getting a valid proxy signature $\sigma_p = \{Y_0, Y_1, Y_2, Y_3, Y_4\}$, A can forge another identity-based proxy signature $\sigma_p' = \{Y_0', Y_1', Y_2', Y_3', Y_4'\}$ as follows.

(a) A sets ID_{a2} as the original signer and u^* as the proxy signer. A produces a valid warrant m_w'. A produces a delegation $(T_{a2,0}, T_{a2,1}, T_{a2,2})$ on (ID_{a2}, u^*, m_w').

(b) A computes $Y_0' = Y_0 \cdot (T_{a1,0})^{-1} \cdot T_{a2,0}$, $Y_1' = Y_1$, $Y_2' = T_{a2,1}$, $Y_3' = T_{a2,2}$, $Y_4' = Y_4$.

(c) A outputs $\sigma_p' = \{Y_0', Y_1', Y_2', Y_3', Y_4'\}$ as the forged proxy signature, where ID_{a2} is the original signer, u^* is the proxy signer, m_w' is the warrant and m is the message.

It can be verified that $\sigma_p' = \{Y_0', Y_1', Y_2', Y_3', Y_4'\}$ is a valid proxy signature. First we have $e(Y_0', g) =$

$e(Y_0 \cdot (T_{a1,0})^{-1} \cdot T_{a2,0}, g) = e(Y_0, g) \cdot e((T_{a1,0})^{-1}, g) \cdot e(T_{a2,0}, g)$. As

$$e(Y_0, g) = e(g_2, g_1)^2 \cdot e(\mu \cdot \mu_0^{H(u^*)}, Y_1) \cdot e(\mu \cdot \mu_0^{H(ID_{a1})}, Y_2) \cdot e(v \cdot v_0^{H(m_w)}, Y_3) \cdot e(\varpi, Y_4) \cdot e(\tau, Y_4^{H(m)})$$
$$= e(g_2, g_1)^2 \cdot e(\mu \cdot \mu_0^{H(u^*)}, Y_1') \cdot e(\mu \cdot \mu_0^{H(ID_{a1})}, Y_2) \cdot e(v \cdot v_0^{H(m_w)}, Y_3) \cdot e(\varpi, Y_4') \cdot e(\tau, (Y_4')^{H(m)}),$$
$$e((T_{a1,0})^{-1}, g) = e(g_2, g_1)^{-1} \cdot e(\mu \cdot \mu_0^{H(ID_{a1})}, T_{a1,1})^{-1} \cdot e(v \cdot v_0^{H(m_w)}, T_{a1,2})^{-1}$$
$$= e(g_2, g_1)^{-1} \cdot e(\mu \cdot \mu_0^{H(ID_{a1})}, Y_2)^{-1} \cdot e(v \cdot v_0^{H(m_w)}, Y_3)^{-1}$$
$$e(T_{a2,0}, g) = e(g_2, g_1) \cdot e(\mu \cdot \mu_0^{H(ID_{a2})}, T_{a2,1}) \cdot e(v \cdot v_0^{H(m_w')}, T_{a2,2})$$
$$= e(g_2, g_1) \cdot e(\mu \cdot \mu_0^{H(ID_{a2})}, Y_2') \cdot e(v \cdot v_0^{H(m_w')}, Y_3'),$$ we have
$$e(Y_0', g) = e(g_2, g_1)^2 \cdot e(\mu \cdot \mu_0^{H(u^*)}, Y_1') \cdot e(\mu \cdot \mu_0^{H(ID_{a2})}, Y_2') \cdot e(v \cdot v_0^{H(m_w')}, Y_3') \cdot e(\varpi, Y_4') \cdot e(\tau, (Y_4')^{H(m)}),$$ and
$\sigma_p' = \{Y_0', Y_1', Y_2', Y_3', Y_4'\}$ is valid.

4.2 Forging a Proxy Signature 2

In the following, we will show that after getting a valid standard signature, the adversary A can modify it to a $T = type3$ proxy signature.

As described in Section 4.1, it assumes that user u^* is not corrupted while user ID_{a1} is by the adversary A, that is, the private key $(x_{ID_{a1},0}, x_{ID_{a1},1})$ of ID_{a1} is known by A.

(1) In the signature queries stage, A produces a message m and makes a $T = type1$ (sign-oracle()) oracle query, where u^* is the signer. The challenger C returns a standard signature $\sigma = (X_0, X_1, X_2)$ to A.

(2) Forgery: after getting a valid standard signature $\sigma = (X_0, X_1, X_2)$ of u^*, A can modify it to a proxy signature $\sigma_p' = \{Y_0', Y_1', Y_2', Y_3', Y_4'\}$ as follows.

(a) A sets ID_{a1} as the original signer and u^* as the proxy signer. A produces a warrant m_w'. A produces a delegation $(T_{a1,0}, T_{a1,1}, T_{a1,2})$ on (ID_{a1}, u^*, m_w').

(b) A computes $Y_0' = X_0 \cdot T_{a1,0}$, $Y_1' = X_1$, $Y_2' = T_{a1,1}$, $Y_3' = T_{a1,2}$, $Y_4' = X_2$.

(c) A outputs $\sigma_p' = \{Y_0', Y_1', Y_2', Y_3', Y_4'\}$ as the forged proxy signature, where ID_{a1} is the original signer, u^* is the proxy signer, m_w' is the warrant and m is the message.

It can be verified that $\sigma_p' = \{Y_0', Y_1', Y_2', Y_3', Y_4'\}$ is a valid proxy signature. First we have $e(Y_0', g) = e(X_0 \cdot T_{a1,0}, g) = e(X_0, g) \cdot e(T_{a1,0}, g)$. As

$$e(X_0, g) = e(g_2, g_1) \cdot e(\mu \cdot \mu_0^{H(u^*)}, X_1) \cdot e(\varpi, X_2) \cdot e(\tau, X_2^{H(m)}) =$$
$$e(g_2, g_1) \cdot e(\mu \cdot \mu_0^{H(u^*)}, Y_1') \cdot e(\varpi, Y_4') \cdot e(\tau, (Y_4')^{H(m)}),$$
$$e(T_{a1,0}, g) = e(g_2, g_1) \cdot e(\mu \cdot \mu_0^{H(ID_{a1})}, T_{a1,1}) \cdot e(v \cdot v_0^{H(m_w)}, T_{a1,2}) =$$
$$e(g_2, g_1) \cdot e(\mu \cdot \mu_0^{H(ID_{a1})}, Y_2') \cdot e(v \cdot v_0^{H(m_w)}, Y_3'),$$ we have
$$e(Y_0', g) = e(g_2, g_1)^2 \cdot e(\mu \cdot \mu_0^{H(u^*)}, Y_1') \cdot e(\mu \cdot \mu_0^{H(ID_{a1})}, Y_2') \cdot e(v \cdot v_0^{H(m_w')}, Y_3') \cdot e(\varpi, Y_4') \cdot e(\tau, (Y_4')^{H(m)}),$$ and
$\sigma_p' = \{Y_0', Y_1', Y_2', Y_3', Y_4'\}$ is valid.

4.3 Forging a Standard Signature

In the following, we will show that after getting a valid $T = type3$ proxy signature, the adversary A can modify it to a standard signature.

As described in Section 4.1, it assumes that user u^* is not corrupted while user ID_{a1} is by the adversary A, that is, the private key $(x_{ID_{a1},0}, x_{ID_{a1},1})$ of ID_{a1} is known by A.

(1) A sets ID_{a1} as the original signer and u^* as the proxy signer. A produces a warrant m_w. A produces a delegation $(T_{a1,0}, T_{a1,1}, T_{a1,2})$ on (ID_{a1}, u^*, m_w).

(2) In the signature queries stage, A makes a $T = type3$ (psign-oracle2()) oracle query, where ID_{a1} is the original signer, u^* is the proxy signer, m_w is the warrant and m is the message. The challenger C returns a proxy signature $\sigma_p = \{Y_0, Y_1, Y_2, Y_3, Y_4\}$ to A.

(3) Forgery: After getting a valid proxy signature $\sigma_p = \{Y_0, Y_1, Y_2, Y_3, Y_4\}$, A can modify it to a standard signature $\sigma' = \{X_0', X_1', X_2'\}$ as follows.

(a) A computes $X_0' = Y_0 \cdot (T_{a1,0})^{-1}$, $X_1' = Y_1$, $X_2' = Y_4$.

(b) A outputs $\sigma' = \{X_0', X_1', X_2'\}$ as the forged standard signature, where u^* is the signer and m is the message.

It can be verified that $\sigma' = \{X_0', X_1', X_2'\}$ is a valid standard signature. First we have $e(X_0', g) = e(Y_0 \cdot (T_{a1,0})^{-1}, g) = e(Y_0, g) \cdot e((T_{a1,0})^{-1}, g)$. As

$$e(Y_0, g) = e(g_2, g_1)^2 \cdot e(\mu \cdot \mu_0^{H(u^*)}, Y_1) \cdot e(\mu \cdot \mu_0^{H(ID_{a1})}, Y_2) \cdot e(v \cdot v_0^{H(m_w)}, Y_3) \cdot e(\varpi, Y_4) \cdot e(\tau, Y_4^{H(m)})$$
$$= e(g_2, g_1)^2 \cdot e(\mu \cdot \mu_0^{H(u^*)}, X_1') \cdot e(\mu \cdot \mu_0^{H(ID_{a1})}, Y_2) \cdot e(v \cdot v_0^{H(m_w)}, Y_3) \cdot e(\varpi, X_2') \cdot e(\tau, (X_2')^{H(m)}),$$
$$e((T_{a1,0})^{-1}, g) = e(g_2, g_1)^{-1} \cdot e(\mu \cdot \mu_0^{H(ID_{a1})}, T_{a1,1})^{-1} \cdot e(v \cdot v_0^{H(m_w)}, T_{a1,2})^{-1}$$
$$= e(g_2, g_1)^{-1} \cdot e(\mu \cdot \mu_0^{H(ID_{a1})}, Y_2)^{-1} \cdot e(v \cdot v_0^{H(m_w)}, Y_3)^{-1},$$
we have $e(X_0', g) = e(g_2, g_1) \cdot e(\mu \cdot \mu_0^{H(u^*)}, X_1') \cdot e(\varpi, X_2') \cdot e(\tau, (X_2')^{H(m)})$ and $\sigma' = \{X_0', X_1', X_2'\}$ is valid.

4.4 Proxy Key Exposure Attack

As noted in scheme [19], proxy key is often used in a hostile environment like mobile agent. Compromised proxy key must not leak information about the long-term private key. In scheme [19], Schuldt et al. introduced the proxy key exposure attack. In Hu et al.'s security model, they also considered this type of attack, that is, an attacker can get the proxy-key oracle service. In the following, we will show that after getting a valid $T = type3$ proxy key, the adversary A can compute the private key of the proxy signer.

As described in Section 4.1, it assumes that user u^* is not corrupted while user ID_{a1} is by the adversary A, that is, the private key $(x_{ID_{a1},0}, x_{ID_{a1},1})$ of ID_{a1} is known by A.

(1) A sets ID_{a1} as the original signer and u^* as the proxy signer. A produces a valid warrant m_w. A produces a delegation $(T_{a1,0}, T_{a1,1}, T_{a1,2})$ on (ID_{a1}, u^*, m_w).

(2) In the key queries stage, the adversary A makes a $T = type3$ (proxykey-oracle2()) oracle query. The challenger C returns the proxy key $PSK_{u^*} = (y_0, y_1, y_2, y_3)$ of user u^*.

(3) Now, the adversary A can compute $x_{u^*,0} = y_0 \cdot (T_{a1,0})^{-1}$, $x_{u^*,1} = y_1$. Thus, the adversary A can get the private key $sk_{u^*} = (x_{u^*,0}, x_{u^*,1})$ of the proxy signer u^*.

5 An Improved Scheme

Why Hu et al.'s improved scheme is not secure? Intuitively, let us see the IProxySign algorithm. $Y_0 = y_0 \cdot \varpi^d \cdot \tau^{d \cdot H(m)} = x_{ID_p,0} \cdot T_0 \cdot \varpi^d \cdot \tau^{d \cdot H(m)}$, where the T_0 part is independent of other parameters. Thus a malicious original signer who knows the T_0 can remove it from Y_0 by multiplying $(T_0)^{-1}$ to get a standard signature of ID_p. Second, if he/she get a standard signature of ID_p, he/she can also multiply T_0 to get a proxy signature. Third, from Y_0, if he/she substitutes T_0 with T_0', he/she can forge a proxy signature where the original signer is ID_a'.

In addition, the proxy key in Hu et al.'s scheme is $y_0 = x_{ID_p,0} \cdot T_0$. Thus, a malicious original signer can remove T_0 by multiplying $(T_0)^{-1}$ to get the private key of ID_p. This kind of attack is named as the proxy key exposure attack, which is introduced by Schuldt et al. [19].

Our improvements are mainly focused on the hash function. In the KeyGen algorithm, we use $H(ID, x_{ID,0})$ instead of $H(ID)$. In the ISign algorithm, we use $H(m, X_0, X_1)$ instead of $H(m)$. In the IDelegate algorithm, we use $H(m_w, T_0, T_1)$ instead of $H(m_w)$. In the IProxyKeyGen algorithm, we add a random number $t \in Z_q^*$. In the IProxySign algorithm, we use $H(m, m_w, y_0, y_1, y_2, y_3, y_4)$ instead of $H(m)$. After making these changes, our improved scheme can resist Hu et al's attack [18] and our attacks.

Setup. Given a security parameter 1^k, the PKG chooses two cyclic multiplicative groups G_1 and G_2 of prime order q, a random generator g of G_1, a bilinear map: $e: G_1 \times G_1 \to G_2$ and a hash function $H: \{0,1\}^* \to Z_q^*$. The PKG randomly chooses $\alpha \in Z_q^*$ and $g_2, \mu, \mu_0, v, \tau \in G_1^*$. The PKG sets $g_1 = g^\alpha$. The system public parameters are $Params = \{G_1, G_2, e, g, g_1, g_2, \mu, \mu_0, v, \tau, H\}$. The system master private key is α.

KeyGen. Given an identity ID, the PKG randomly chooses $r_{ID} \in Z_q^*$ and computes $x_{ID,0} = g_1^{r_{ID}}$ and $x_{ID,1} = g_2^\alpha \cdot (\mu \cdot \mu_0^{H(ID, x_{ID,0})})^{\alpha \cdot r_{ID}}$. The private key of user ID is $sk_{ID} = (x_{ID,0}, x_{ID,1})$. Then the original signer ID_a 's private key is $sk_{ID_a} = (x_{ID_a,0}, x_{ID_a,1}) = (g_1^{r_{ID_a}}, g_2^\alpha \cdot (\mu \cdot \mu_0^{H(ID_a, x_{ID_a,0})})^{\alpha \cdot r_{ID_a}})$ and the proxy signer ID_p 's private key is $sk_{ID_p} = (x_{ID_p,0}, x_{ID_p,1}) = (g_1^{r_{ID_p}}, g_2^\alpha \cdot (\mu \cdot \mu_0^{H(ID_p, x_{ID_p,0})})^{\alpha \cdot r_{ID_p}})$.

ISign. Given an identity ID and a message m, the signer randomly chooses $d \in Z_q^*$ and computes $X_0 = x_{ID,0} = g_1^{r_{ID}}$, $X_1 = g^d$, $X_2 = x_{ID,1} \cdot \tau^{d \cdot H(m, X_0, X_1)}$. Finally, the signature is $\sigma = (m, X_0, X_1, X_2)$.

Iverify. The verifier checks whether the following equation holds. $e(X_2, g) = e(g_2, g_1) \cdot e(\mu \cdot \mu_0^{H(ID, X_0)}, X_0) \cdot e(\tau^{H(m, X_0, X_1)}, X_1)$.

IDelegate. The original signer ID_a produces a warrant m_w, which contains the descriptions of delegation duration, delegation message type, the identities of original signer and proxy signer and so on. Then he/she randomly chooses $s \in Z_q^*$ and computes $T_0 = x_{ID_a,0} = g_1^{r_{ID_a}}$, $T_1 = g^s$, $T_2 = x_{ID_a,1} \cdot v^{H(m_w, T_0, T_1) \cdot s}$. Finally, the delegation is $\delta = (m_w, T_0, T_1, T_2)$.

IProxyKeyGen. The proxy signer ID_p first checks whether the delegation is valid by checking the following equation. $e(T_2, g) = e(g_2, g_1) \cdot e(\mu \cdot \mu_0^{H(ID_a, T_0)}, T_0) \cdot e(v^{H(m_w, T_0, T_1)}, T_1)$. If it is not true, then he/she requests the original signer to reproduce the delegation $\delta = (m_w, T_0, T_1, T_2)$. Otherwise, he/she randomly chooses $t \in Z_q^*$ and computes $y_0 = x_{ID_p,0} = g_1^{r_{ID_p}}$, $y_1 = T_0 = g_1^{r_{ID_a}}$,

$y_2 = T_1 = g^s$, $y_3 = g^t$, $y_4 = x_{ID_p,1} \cdot v^{H(m_w, y_0, y_1, y_2, y_3) \cdot t} \cdot T_2$. Finally, the proxy key is $PSK_{ID_p} = (y_0, y_1, y_2, y_3, y_4)$.

IProxySign. Given a message m, the proxy signer ID_p randomly chooses $d \in Z_q^*$ and computes $Y_0 = y_0 = g_1^{r_{ID_p}}$, $Y_1 = y_1 = g_1^{r_{ID_a}}$, $Y_2 = y_2$, $Y_3 = y_3$, $Y_4 = g^d$, $Y_5 = y_4 \cdot \tau^{d \cdot H(m, m_w, Y_0, Y_1, Y_2, Y_3, Y_4)}$. Finally, the proxy signature is $\sigma_p = (m, m_w, Y_0, Y_1, Y_2, Y_3, Y_4, Y_5)$.

IProxyVerify. The verifier checks whether the following equation holds. $e(Y_5, g) = e(g_2, g_1)^2 \cdot e(\mu \cdot \mu_0^{H(ID_p, Y_0)}, Y_0) \cdot e(\mu \cdot \mu_0^{H(ID_a, Y_1)}, Y_1) \cdot e(v^{H(m_w, Y_0, Y_1, Y_2, Y_3)}, Y_3) \cdot e(v^{H(m_w, Y_1, Y_2)}, Y_2) \cdot e(\tau^{H(m, m_w, Y_0, Y_1, Y_2, Y_3, Y_4)}, Y_4)$.

6 Analysis of the Improved Scheme

6.1 Correctness

$$
\begin{aligned}
&e(Y_5, g) = e(y_4 \cdot \tau^{d \cdot H(m, m_w, Y_0, Y_1, Y_2, Y_3, Y_4)}, g) \\
&= e(x_{ID_p,1} \cdot v^{H(m_w, y_0, y_1, y_2, y_3) \cdot t} \cdot T_2, g) \cdot e(\tau^{H(m, m_w, Y_0, Y_1, Y_2, Y_3, Y_4)}, Y_4) \\
&= e(x_{ID_p,1}, g) \cdot e(v^{H(m_w, y_0, y_1, y_2, y_3) \cdot t}, g) \cdot e(T_2, g) \\
&\quad \cdot e(\tau^{H(m, m_w, Y_0, Y_1, Y_2, Y_3, Y_4)}, Y_4) \\
&= e(x_{ID_p,1}, g) \cdot e(v^{H(m_w, y_0, y_1, y_2, y_3) \cdot t}, g) \cdot e(x_{ID_a,1} \cdot v^{H(m_w, T_0, T_1) \cdot s}, g) \\
&\quad \cdot e(\tau^{H(m, m_w, Y_0, Y_1, Y_2, Y_3, Y_4)}, Y_4) \\
&= e(g_2, g_1) \cdot e(\mu \cdot \mu_0^{H(ID_p, Y_0)}, Y_0) \cdot e(v^{H(m_w, y_0, y_1, y_2, y_3) \cdot t}, g) \\
&\quad \cdot e(x_{ID_a,1} \cdot v^{H(m_w, T_0, T_1) \cdot s}, g) \cdot e(\tau^{H(m, m_w, Y_0, Y_1, Y_2, Y_3, Y_4)}, Y_4) \\
&= e(g_2, g_1)^2 \cdot e(\mu \cdot \mu_0^{H(ID_p, Y_0)}, Y_0) \cdot e(\mu \cdot \mu_0^{H(ID_a, Y_1)}, Y_1) \\
&\quad \cdot e(v^{H(m_w, y_0, y_1, y_2, y_3)}, Y_3) \cdot e(v^{H(m_w, T_0, T_1)}, Y_2) \\
&\quad \cdot e(\tau^{H(m, m_w, Y_0, Y_1, Y_2, Y_3, Y_4)}, Y_4) \\
&= e(g_2, g_1)^2 \cdot e(\mu \cdot \mu_0^{H(ID_p, Y_0)}, Y_0) \cdot e(\mu \cdot \mu_0^{H(ID_a, Y_1)}, Y_1) \\
&\quad \cdot e(v^{H(m_w, Y_0, Y_1, Y_2, Y_3)}, Y_3) \cdot e(v^{H(m_w, Y_1, Y_2)}, Y_2) \\
&\quad \cdot e(\tau^{H(m, m_w, Y_0, Y_1, Y_2, Y_3, Y_4)}, Y_4)
\end{aligned}
$$

6.2 Unforgeability

Theorem 1. Our improved scheme is $(\varepsilon, t, q_e, q_s)-$ secure, assuming that the $(\varepsilon', t')-$ CDH assumption holds in G_1, where $\varepsilon' = 3\varepsilon / q$, $t' = t + O(q_e \cdot (11 \cdot t_{mul} + 15 \cdot t_{\exp}) + q_s \cdot (4 \cdot t_{mul} + 5 \cdot t_{\exp}))$, and q_e, q_s, $t_{\exp}$ and t_{mul} are the maximal number of private key queries, signature queries, the time required for an exponentiation and an multiplication in G_1, respectively.

Proof. Our proof is similar to that of Hu et al.'s scheme [18]. Let A be an $(\varepsilon, t, q_e, q_s)-$ adversary attacking our improved scheme. From this adversary, we will construct an algorithm C that solves the CDH problem with a probability of at least ε' and in the time of at most t', contradicting the $(\varepsilon', t')-$ CDH assumption.

C is given $(g, g^a, g^b) \in G_1$ for randomly chosen $a, b \in Z_q^*$. C does not know the values of a and b, and is asked to compute g^{ab}. To utilize the adversary A, C simulates all the oracles defined in Definition 1 to provide responses to A's queries.

Setup. C randomly chooses two cyclic groups G_1 and G_2 of prime order q, a random generator g of G_1, a bilinear map $e: G_1 \times G_1 \to G_2$ and a hash function $H: \{0,1\}^* \to Z_q^*$. He/she sets $g_1 = g^a$ and $g_2 = g^b$. He/she randomly chooses $\gamma, \eta, \lambda, \theta \in Z_q^*$ and sets $\mu = g_2^\gamma \cdot g$, $v = g^\eta$, $\tau = g_2^\lambda \cdot g$ and $\mu_0 = g^\theta$. C outputs the public parameters $Params = \{G_1, G_2, e, g, g_1, g_2, \mu, \mu_0, v, \tau, H\}$. The system master private key is a, which is not known to C.

We assume that user 1 is a non-corrupted user and ID_1 is his/her identity. C first computes the private key of ID_1. C randomly chooses $r_{ID_1} \in Z_q^*$ and computes $x_{ID_1,0} = g_1^{-1/\gamma} \cdot g^{r_{ID_1}}$, $x_{ID_1,1} = g_1^{-(1+H(ID_1, x_{ID_1,0}) \cdot \theta)/\gamma} \cdot \mu^{r_{ID_1}} \cdot \mu_0^{r_{ID_1} \cdot H(ID_1, x_{ID_1,0})}$. Then, the private key of ID_1 is $sk_{ID_1} = (x_{ID_1,0}, x_{ID_1,1})$, which is a valid private key because

$$
\begin{aligned}
&x_{ID_1,0} = g_1^{-1/\gamma} \cdot g^{r_{ID_1}} = g_1^{-1/\gamma} \cdot g^{r_{ID_1} \cdot a/a} = g_1^{r_{ID_1}/a - 1/\gamma} = g_1^{r'_{ID_1}}, \\
&x_{ID_1,1} = g_1^{-(1+H(ID_1, x_{ID_1,0}) \cdot \theta)/\gamma} \cdot \mu^{r_{ID_1}} \cdot \mu_0^{r_{ID_1} \cdot H(ID_1, x_{ID_1,0})} \\
&= g_2^a \cdot g_2^{-a} \cdot g_1^{-1/\gamma} \cdot g_1^{-H(ID_1, x_{ID_1,0}) \cdot \theta/\gamma} \cdot \mu^{r_{ID_1}} \cdot \mu_0^{r_{ID_1} \cdot H(ID_1, x_{ID_1,0})} \\
&= g_2^a \cdot (g_2^\gamma \cdot g)^{-a/\gamma} \cdot g_1^{-H(ID_1, x_{ID_1,0}) \cdot \theta/\gamma} \cdot \mu^{r_{ID_1}} \cdot \mu_0^{r_{ID_1} \cdot H(ID_1, x_{ID_1,0})} \\
&= g_2^a \cdot (g_2^\gamma \cdot g)^{-a/\gamma} \cdot \mu_0^{-H(ID_1, x_{ID_1,0}) \cdot a/\gamma} \cdot \mu^{r_{ID_1}} \cdot \mu_0^{r_{ID_1} \cdot H(ID_1, x_{ID_1,0})} \\
&= g_2^a \cdot \mu^{-a/\gamma} \cdot \mu_0^{-H(ID_1, x_{ID_1,0}) \cdot a/\gamma} \cdot \mu^{r_{ID_1}} \cdot \mu_0^{r_{ID_1} \cdot H(ID_1, x_{ID_1,0})} \\
&= g_2^a \cdot \mu^{a \cdot (r_{ID_1}/a - 1/\gamma)} \cdot \mu_0^{a \cdot (r_{ID_1}/a - 1/\gamma) \cdot H(ID_1, x_{ID_1,0})} \\
&= g_2^a \cdot (\mu \cdot \mu_0^{H(ID_1, x_{ID_1,0})})^{a \cdot (r_{ID_1}/a - 1/\gamma)} \\
&= g_2^a \cdot (\mu \cdot \mu_0^{H(ID_1, x_{ID_1,0})})^{a \cdot r'_{ID_1}}, \text{ where } r'_{ID_1} = r_{ID_1}/a - 1/\gamma.
\end{aligned}
$$

C keeps $sk_{ID_1} = (x_{ID_1,0}, x_{ID_1,1})$ private.

Queries. A can adaptively make a polynomially bounded number of queries as follows.

Key queries.

(1) $T = type1$: A requests the private key of ID_i $(i > 1)$. A supplies an identity ID_i $(i > 1)$. C produces ID_i's private key as described above, that is, C randomly chooses $r_{ID_i} \in Z_q^*$ and computes $x_{ID_i,0} = g_1^{-1/\gamma} \cdot g^{r_{ID_i}}$, $x_{ID_i,1} = g_1^{-(1+H(ID_i, x_{ID_i,0}) \cdot \theta)/\gamma} \cdot \mu^{r_{ID_i}} \cdot \mu_0^{r_{ID_i} \cdot H(ID_i, x_{ID_i,0})}$. Then, the private key of ID_i is $sk_{ID_i} = (x_{ID_i,0}, x_{ID_i,1})$. C returns it to A.

(2) $T = type2$: A requests the proxy key of ID_i ($i \geq 1$), where ID_i ($i \geq 1$) is both the original signer and the proxy signer. A supplies an identity ID_i ($i \geq 1$) and a warrant m_w. C produces the self-delegation proxy key as normal because C can get the private keys of all users.

(3) $T = type3$: A requests the proxy key of ID_1, where ID_i ($i > 1$) is the original signer and ID_1 the proxy signer. A supplies an identity ID_i ($i > 1$) and a warrant m_w. C produces the proxy key as normal because C can get the private keys of ID_i ($i > 1$) and ID_1.

(4) $T = type4$: A requests the proxy key of ID_i ($i > 1$), where ID_1 is the original signer and ID_i ($i > 1$) the proxy signer. A supplies an identity ID_i ($i > 1$) and a warrant m_w. C produces the proxy key as normal because C can get the private keys of ID_i ($i > 1$) and ID_1.

Signature queries.

(1) $T = type1$: A requests a standard signature of ID_i ($i \geq 1$). A supplies an identity ID_i ($i \geq 1$) and a message m. C produces the standard signature as normal because C can get the private keys of ID_i ($i \geq 1$).

(2) $T = type2$: A requests a self-delegation proxy signature of ID_i ($i \geq 1$), where ID_i ($i \geq 1$) is both the original signer and the proxy signer. A supplies an identity ID_i ($i \geq 1$), a warrant m_w and a message m. C produces the self-delegation proxy signature as normal because C can get the private keys of all users.

(3) $T = type3$: A requests a proxy signature of ID_1, where ID_i ($i > 1$) is the original signer and ID_1 the proxy signer. A supplies an identity ID_i ($i > 1$), a warrant m_w and a message m. C produces the proxy signature as normal because C can get the private keys of ID_i ($i > 1$) and ID_1.

(4) $T = type4$: A requests a proxy signature of ID_i ($i > 1$), where ID_1 is the original signer and ID_i ($i > 1$) the proxy signer. A supplies an identity ID_i ($i > 1$), a warrant m_w and a message m. C produces the proxy signature as normal because C can get the private keys of ID_i ($i > 1$) and ID_1.

Forgery. At last, A decides to put an end to the queries stage and outputs a forgery.

(1) $T = type1$: A outputs a forged standard signature $\sigma^* = (m^*, X_0^*, X_1^*, X_2^*)$ of ID^*.. If $a \cdot \gamma \cdot r_{ID^*} + \lambda \cdot d^* \cdot H(m^*, X_0^*, X_1^*) \neq 0 \bmod q$, then C aborts; otherwise, C can compute

$$\frac{X_2^*}{X_0^* \cdot (X_0^*)^{\theta \cdot H(ID^*, X_0^*)} \cdot (X_1^*)^{H(m^*, X_0^*, X_1^*)}}$$

$$= \frac{x_{ID^*,1} \cdot \tau^{d^* \cdot H(m^*, X_0^*, X_1^*)}}{X_0^* \cdot (X_0^*)^{\theta \cdot H(ID^*, X_0^*)} \cdot (X_1^*)^{H(m^*, X_0^*, X_1^*)}}$$

$$= \frac{g_2^a \cdot (\mu \cdot \mu_0^{H(ID^*, x_{ID^*,0})})^{a \cdot r_{ID^*}} \cdot \tau^{d^* \cdot H(m^*, X_0^*, X_1^*)}}{X_0^* \cdot (X_0^*)^{\theta \cdot H(ID^*, X_0^*)} \cdot (X_1^*)^{H(m^*, X_0^*, X_1^*)}}$$

$$= \frac{g_2^a \cdot (g_2^\gamma \cdot g \cdot g^{\theta \cdot H(ID^*, X_0^*)})^{a \cdot r_{ID^*}} \cdot (g_2^\lambda \cdot g)^{d^* \cdot H(m^*, X_0^*, X_1^*)}}{X_0^* \cdot (X_0^*)^{\theta \cdot H(ID^*, X_0^*)} \cdot (X_1^*)^{H(m^*, X_0^*, X_1^*)}}$$

$$= g_2^a = g^{ab}$$

(2) $T = type2$: A outputs a forged self-delegation proxy signature $\sigma_p^* = (m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*, Y_5^*)$ of ID^*. If $2 \cdot a \cdot \gamma \cdot r_{ID^*} + \lambda \cdot d^* \cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*) \neq 0 \bmod q$, then C aborts; otherwise, C can compute

$$\frac{Y_5^*}{(Y_0^*)^2 \cdot (Y_0^*)^{2 \cdot \theta \cdot H(ID^*, Y_0^*)} \cdot (Y_2^*)^{\eta \cdot H(m_w^*, Y_1^*, Y_2^*)} \cdot (Y_3^*)^{\eta \cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}}$$

$$\cdot \frac{1}{(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$

$$= \frac{y_4^* \cdot \tau^{d^* \cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}{(Y_0^*)^2 \cdot (Y_0^*)^{2 \cdot \theta \cdot H(ID^*, Y_0^*)} \cdot (Y_2^*)^{\eta \cdot H(m_w^*, Y_1^*, Y_2^*)} \cdot (Y_3^*)^{\eta \cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}}$$

$$\cdot \frac{1}{(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$

$$= \frac{x_{ID^*,1} \cdot v^{H(m_w^*, y_0^*, y_1^*, y_2^*, y_3^*) \cdot t^*} \cdot T_2^* \cdot \tau^{d^* \cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}{(Y_0^*)^2 \cdot (Y_0^*)^{2 \cdot \theta \cdot H(ID^*, Y_0^*)} \cdot (Y_2^*)^{\eta \cdot H(m_w^*, Y_1^*, Y_2^*)} \cdot (Y_3^*)^{\eta \cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}}$$

$$\cdot \frac{1}{(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$

$$= \frac{g_2^{2 \cdot a} \cdot (\mu \cdot \mu_0^{H(ID^*, x_{ID^*,0})})^{2 \cdot a \cdot r_{ID^*}} \cdot v^{H(m_w^*, y_0^*, y_1^*, y_2^*, y_3^*) \cdot t^*} \cdot v^{H(m_w^*, T_0^*, T_1^*) \cdot s^*}}{(Y_0^*)^2 \cdot (Y_0^*)^{2 \cdot \theta \cdot H(ID^*, Y_0^*)} \cdot (Y_2^*)^{\eta \cdot H(m_w^*, Y_1^*, Y_2^*)} \cdot (Y_3^*)^{\eta \cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}}$$

$$\cdot \frac{\tau^{d^* \cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}{(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$

$$= \frac{g_2^{2 \cdot a} \cdot (g_2^\gamma \cdot g \cdot g^{\theta \cdot H(ID^*, x_{ID^*,0})})^{2 \cdot a \cdot r_{ID^*}} \cdot g^{\eta \cdot H(m_w^*, y_0^*, y_1^*, y_2^*, y_3^*) \cdot t^*}}{(Y_0^*)^2 \cdot (Y_0^*)^{2 \cdot \theta \cdot H(ID^*, Y_0^*)} \cdot (Y_2^*)^{\eta \cdot H(m_w^*, Y_1^*, Y_2^*)} \cdot (Y_3^*)^{\eta \cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}}$$

$$\cdot \frac{g^{\eta \cdot H(m_w^*, T_0^*, T_1^*) \cdot s^*} \cdot (g_2^\lambda \cdot g)^{d^* \cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}{(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$

$$= \frac{g_2^{2 \cdot a} \cdot (g_2^\gamma \cdot g \cdot g^{\theta \cdot H(ID^*, Y_0^*)})^{2 \cdot a \cdot r_{ID^*}} \cdot g^{\eta \cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*) \cdot t^*}}{(Y_0^*)^2 \cdot (Y_0^*)^{2 \cdot \theta \cdot H(ID^*, Y_0^*)} \cdot (Y_2^*)^{\eta \cdot H(m_w^*, Y_1^*, Y_2^*)} \cdot (Y_3^*)^{\eta \cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}}$$

$$\cdot \frac{g^{\eta \cdot H(m_w^*, Y_1^*, Y_2^*) \cdot s^*} \cdot (g_2^\lambda \cdot g)^{d^* \cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}{(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}} = g_2^{2 \cdot a} = g^{2ab},$$

from which C can compute g^{ab}.

(3) $T = type3$: A outputs a forged proxy signature

$\sigma_p^* = (m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*, Y_5^*)$, where ID_i^* ($i>1$) is the original signer and ID_1 is the proxy signer. If $a\cdot\gamma\cdot r_{ID_1} + a\cdot\gamma\cdot r_{ID_i^*} + \lambda\cdot d^*\cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*) \neq 0 \bmod q$, then C aborts; otherwise, C can compute

$$\frac{Y_5^*}{Y_0^*\cdot(Y_0^*)^{\theta\cdot H(ID_1, Y_0^*)}\cdot Y_1^*\cdot(Y_1^*)^{\theta\cdot H(ID_i^*, Y_1^*)}\cdot(Y_2^*)^{\eta\cdot H(m_w^*, Y_1^*, Y_2^*)}}$$
$$\cdot\frac{1}{(Y_3^*)^{\eta\cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}\cdot(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$
$$= \frac{y_4^*\cdot\tau^{d^*\cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}{Y_0^*\cdot(Y_0^*)^{\theta\cdot H(ID_1, Y_0^*)}\cdot Y_1^*\cdot(Y_1^*)^{\theta\cdot H(ID_i^*, Y_1^*)}\cdot(Y_2^*)^{\eta\cdot H(m_w^*, Y_1^*, Y_2^*)}}$$
$$\cdot\frac{1}{(Y_3^*)^{\eta\cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}\cdot(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$
$$= \frac{x_{ID_1,1}\cdot v^{H(m_w^*, y_0^*, y_1^*, y_2^*, y_3^*)\cdot t^*}\cdot T_2^*\cdot\tau^{d^*\cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}{Y_0^*\cdot(Y_0^*)^{\theta\cdot H(ID_1, Y_0^*)}\cdot Y_1^*\cdot(Y_1^*)^{\theta\cdot H(ID_i^*, Y_1^*)}\cdot(Y_2^*)^{\eta\cdot H(m_w^*, Y_1^*, Y_2^*)}}$$
$$\cdot\frac{1}{(Y_3^*)^{\eta\cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}\cdot(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$
$$= \frac{g_2^a\cdot(\mu\cdot\mu_0^{H(ID_1, x_{ID_1,0})})^{a\cdot r_{ID_1}}\cdot v^{H(m_w^*, y_0^*, y_1^*, y_2^*, y_3^*)\cdot t^*}}{Y_0^*\cdot(Y_0^*)^{\theta\cdot H(ID_1, Y_0^*)}\cdot Y_1^*\cdot(Y_1^*)^{\theta\cdot H(ID_i^*, Y_1^*)}\cdot(Y_2^*)^{\eta\cdot H(m_w^*, Y_1^*, Y_2^*)}}$$
$$\cdot\frac{g_2^a\cdot(\mu\cdot\mu_0^{H(ID_i^*, x_{ID_i*,0})})^{a\cdot r_{ID_i^*}}\cdot v^{H(m_w^*, T_0^*, T_1^*)\cdot s^*}\cdot\tau^{d^*\cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}{(Y_3^*)^{\eta\cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}\cdot(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$
$$= \frac{g_2^a\cdot(g_2^\gamma\cdot g\cdot g^{\theta\cdot H(ID_1, x_{ID_1,0})})^{a\cdot r_{ID_1}}\cdot g^{\eta\cdot H(m_w^*, y_0^*, y_1^*, y_2^*, y_3^*)\cdot t^*}}{Y_0^*\cdot(Y_0^*)^{\theta\cdot H(ID_1, Y_0^*)}\cdot Y_1^*\cdot(Y_1^*)^{\theta\cdot H(ID_i^*, Y_1^*)}\cdot(Y_2^*)^{\eta\cdot H(m_w^*, Y_1^*, Y_2^*)}}$$
$$\cdot\frac{g_2^a\cdot(g_2^\gamma\cdot g\cdot g^{\theta\cdot H(ID_i^*, x_{ID_i*,0})})^{a\cdot r_{ID_i^*}}\cdot g^{\eta\cdot H(m_w^*, T_0^*, T_1^*)\cdot s^*}}{(Y_3^*)^{\eta\cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}\cdot(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$
$$\cdot(g_2^\lambda\cdot g)^{d^*\cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}$$
$$= \frac{g_2^a\cdot(g_2^\gamma\cdot g\cdot g^{\theta\cdot H(ID_1, Y_0^*)})^{a\cdot r_{ID_1}}\cdot g^{\eta\cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)\cdot t^*}}{Y_0^*\cdot(Y_0^*)^{\theta\cdot H(ID_1, Y_0^*)}\cdot Y_1^*\cdot(Y_1^*)^{\theta\cdot H(ID_i^*, Y_1^*)}\cdot(Y_2^*)^{\eta\cdot H(m_w^*, Y_1^*, Y_2^*)}}$$
$$\cdot\frac{g_2^a\cdot(g_2^\gamma\cdot g\cdot g^{\theta\cdot H(ID_i^*, Y_1^*)})^{a\cdot r_{ID_i^*}}\cdot g^{\eta\cdot H(m_w^*, Y_1^*, Y_2^*)\cdot s^*}}{(Y_3^*)^{\eta\cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}\cdot(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$
$$\cdot(g_2^\lambda\cdot g)^{d^*\cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}$$
$$= g_2^{2\cdot a} = g^{2ab}$$, from which C can compute g^{ab}.

(4) $T = type4$: A outputs a forged proxy signature $\sigma_p^* = (m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*, Y_5^*)$, where ID_1 is the original signer and ID_i^* ($i>1$) is the proxy signer. If $a\cdot\gamma\cdot r_{ID_1} + a\cdot\gamma\cdot r_{ID_i^*} + \lambda\cdot d^*\cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*) \neq 0 \bmod q$, then C aborts; otherwise, C can compute

$$\frac{Y_5^*}{Y_0^*\cdot(Y_0^*)^{\theta\cdot H(ID_i^*, Y_0^*)}\cdot Y_1^*\cdot(Y_1^*)^{\theta\cdot H(ID_1, Y_1^*)}\cdot(Y_2^*)^{\eta\cdot H(m_w^*, Y_1^*, Y_2^*)}}$$
$$\cdot\frac{1}{(Y_3^*)^{\eta\cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}\cdot(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$
$$= \frac{y_4^*\cdot\tau^{d^*\cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}{Y_0^*\cdot(Y_0^*)^{\theta\cdot H(ID_i^*, Y_0^*)}\cdot Y_1^*\cdot(Y_1^*)^{\theta\cdot H(ID_1, Y_1^*)}\cdot(Y_2^*)^{\eta\cdot H(m_w^*, Y_1^*, Y_2^*)}}$$
$$\cdot\frac{1}{(Y_3^*)^{\eta\cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}\cdot(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$
$$= \frac{x_{ID_i^*,1}\cdot v^{H(m_w^*, y_0^*, y_1^*, y_2^*, y_3^*)\cdot t^*}\cdot T_2^*\cdot\tau^{d^*\cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}{Y_0^*\cdot(Y_0^*)^{\theta\cdot H(ID_i^*, Y_0^*)}\cdot Y_1^*\cdot(Y_1^*)^{\theta\cdot H(ID_1, Y_1^*)}\cdot(Y_2^*)^{\eta\cdot H(m_w^*, Y_1^*, Y_2^*)}}$$
$$\cdot\frac{1}{(Y_3^*)^{\eta\cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}\cdot(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$
$$= \frac{g_2^a\cdot(\mu\cdot\mu_0^{H(ID_i^*, x_{ID_i*,0})})^{a\cdot r_{ID_i^*}}\cdot v^{H(m_w^*, y_0^*, y_1^*, y_2^*, y_3^*)\cdot t^*}}{Y_0^*\cdot(Y_0^*)^{\theta\cdot H(ID_i^*, Y_0^*)}\cdot Y_1^*\cdot(Y_1^*)^{\theta\cdot H(ID_1, Y_1^*)}\cdot(Y_2^*)^{\eta\cdot H(m_w^*, Y_1^*, Y_2^*)}}$$
$$\cdot\frac{g_2^a\cdot(\mu\cdot\mu_0^{H(ID_1, x_{ID_1,0})})^{a\cdot r_{ID_1}}\cdot v^{H(m_w^*, T_0^*, T_1^*)\cdot s^*}\cdot\tau^{d^*\cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}{(Y_3^*)^{\eta\cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}\cdot(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$
$$= \frac{g_2^a\cdot(g_2^\gamma\cdot g\cdot g^{\theta\cdot H(ID_i^*, x_{ID_i*,0})})^{a\cdot r_{ID_i^*}}\cdot g^{\eta\cdot H(m_w^*, y_0^*, y_1^*, y_2^*, y_3^*)\cdot t^*}}{Y_0^*\cdot(Y_0^*)^{\theta\cdot H(ID_i^*, Y_0^*)}\cdot Y_1^*\cdot(Y_1^*)^{\theta\cdot H(ID_1, Y_1^*)}\cdot(Y_2^*)^{\eta\cdot H(m_w^*, Y_1^*, Y_2^*)}}$$
$$\cdot\frac{g_2^a\cdot(g_2^\gamma\cdot g\cdot g^{\theta\cdot H(ID_1, x_{ID_1,0})})^{a\cdot r_{ID_1}}\cdot g^{\eta\cdot H(m_w^*, T_0^*, T_1^*)\cdot s^*}}{(Y_3^*)^{\eta\cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}\cdot(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$
$$\cdot(g_2^\lambda\cdot g)^{d^*\cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}$$
$$= \frac{g_2^a\cdot(g_2^\gamma\cdot g\cdot g^{\theta\cdot H(ID_i^*, Y_0^*)})^{a\cdot r_{ID_i^*}}\cdot g^{\eta\cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)\cdot t^*}}{Y_0^*\cdot(Y_0^*)^{\theta\cdot H(ID_i^*, Y_0^*)}\cdot Y_1^*\cdot(Y_1^*)^{\theta\cdot H(ID_1, Y_1^*)}\cdot(Y_2^*)^{\eta\cdot H(m_w^*, Y_1^*, Y_2^*)}}$$
$$\cdot\frac{g_2^a\cdot(g_2^\gamma\cdot g\cdot g^{\theta\cdot H(ID_1, Y_1^*)})^{a\cdot r_{ID_1}}\cdot g^{\eta\cdot H(m_w^*, Y_1^*, Y_2^*)\cdot s^*}}{(Y_3^*)^{\eta\cdot H(m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*)}\cdot(Y_4^*)^{H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}}$$
$$\cdot(g_2^\lambda\cdot g)^{d^*\cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*)}$$
$$= g_2^{2\cdot a} = g^{2ab}$$, from which C can compute g^{ab}.

Now we assess the probability of success. In the forgery stage, it must have $a\cdot\gamma\cdot r_{ID^*} + \lambda\cdot d^*\cdot H(m^*, X_0^*, X_1^*) = 0 \bmod q$ for $T = type1$, or $2\cdot a\cdot\gamma\cdot r_{ID^*} + \lambda\cdot d^*\cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*) = 0 \bmod q$ for $T = type2$, or $a\cdot\gamma\cdot r_{ID_1} + a\cdot\gamma\cdot r_{ID_i^*} + \lambda\cdot d^*\cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*) = 0 \bmod q$ for $T = type3$, or $a\cdot\gamma\cdot r_{ID_1} + a\cdot\gamma\cdot r_{ID_i^*} + \lambda\cdot d^*\cdot H(m^*, m_w^*, Y_0^*, Y_1^*, Y_2^*, Y_3^*, Y_4^*) = 0 \bmod q$ for $T = type4$. The equations are the same for $T = type3$ and $T = type4$. The probability of the above four equations holding is all $1/q$. Thus, the total probability is $3\varepsilon/q$.

The time complexity of C depends on the exponentiations and multiplications needed in all above queries. The key queries need 11 t_{mul} computations and 15 $t_{\exp}$ computations. The signature queries need 4 t_{mul} computations and 5 $t_{\exp}$ computations.

6.3 Efficiency

We compare our scheme in terms of computational costs and communicational overheads with other identity-based proxy signature schemes, which include Gu et al.'s scheme [11] and Hu et al.'s scheme [17-18] in the standard model, Shim's scheme [21] and Wu et al.'s scheme [22] in the random oracle model. The comparisons are listed in Table 1 and Table 2. We use e_1 and P to denote a scalar multiplication (or an exponentiation) computation on G_1 and a pairing computation, respectively. Other computations are ignored here as they are not time consuming. $|G_1|$, $|m_w|$, $|q|$ and $|m|$ denote the bit length of an element on G_1, a warrant, the order of G_1 and a message, respectively. "S" and "Rom" denote the standard model and the random oracle model, respectively. The pairing computations that can be precomputed are not included in Table 1. According to scheme [23], a pairing computation is almost 20 times that of a scalar multiplication computation on G_1, so we mainly focus on the pairing computations. From Table 1, we can see that the differences between computation costs in all stages except the proxy verification are one or two e_1 operations, so we can conclude that the computation costs in these stages are very close. In the proxy verification stage, schemes [17] and [21] are the most efficient ones as they only need three pairings. Scheme [18] and ours are the least efficient ones as they need six pairings. From Table 1, we can also conclude that we can make efficient schemes in the random oracle model. Of course, we can also make efficient schemes in the standard model like scheme [17], but the cost is the longer public parameters in the setup stage. Scheme [17] needs n+3 public parameters in the setup stage. In practice, n should be at least 160. Therefore, scheme [17] will need more storage space. Based on the 80-bit security level, $|q|$=160 and $|G_1|$=1024. It will need extra storage space of 160k bits. From Table 2, we can see that schemes [21] and [22] have the shortest length in all aspects. Therefore, schemes in the random oracle model are more communicationally efficient than those in the standard model. Scheme [17] has the shortest length in the standard model. Also based on the 80-bit security level, our scheme is 2752 bits longer than scheme [17] in the proxy signature stage. In general, our scheme increases some computational costs and communicational overheads, but they are still within the acceptable range. Regarding the resistance to proxy key exposure attack, schemes [11, 18, 21] all compute an independent proxy key like ours. But if the proxy key is exposed, the original signer can compute the private key of the proxy signer in all these schemes. About schemes [17, 22], as they do not consider the proxy key exposure attack, they are insecure under this attack.

Table 1. Computational cost and security comparisons with other schemes

Schemes	Delegate	Delegate verify	ProxyKey Gen	Proxy sign	Proxy verify	Public Parameter	Model	Security
Gu et al. [11]	$2e_1$	$2e_1+3P$	0	$3e_1$	$4e_1+5P$	7	S	Insecure
Hu et al. [17]	$2e_1$	e_1+3P	0	$2e_1$	$2e_1+3P$	n+3	S	Secure
Hu et al. [18]	$3e_1$	$2e_1+3P$	0	$3e_1$	$4e_1+6P$	9	S	Insecure
Shim [21]	$3e_1$	e_1+3P	e_1	$3e_1$	$3e_1+3P$	3	Rom	Secure
Wu et al. [22]	$2e_1$	$3P$	0	$2e_1$	$5P$	2	Rom	Secure
Ours	$2e_1$	$2e_1+3P$	$2e_1$	$2e_1$	$5e_1+6P$	7	S	Secure

Note. n denotes the bit length of an identity.

Table 2. Length comparison with other schemes

Schemes	Private key	Delegation	Proxy signature	Against the proxy key exposure attack
Gu et al. [11]	$2\|G_1\|$	$3\|G_1\|+\|m_w\|$	$5\|G_1\|+\|m_w\|+\|m\|$	No
Hu et al. [17]	$\|G_1\|+\|q\|$	$2\|G_1\|+\|q\|$	$3\|G_1\|+2\|q\|+\|m_w\|+\|m\|$	No
Hu et al. [18]	$2\|G_1\|$	$3\|G_1\|+\|m_w\|$	$5\|G_1\|+\|m_w\|+\|m\|$	No
Shim [21]	$\|G_1\|$	$2\|G_1\|+\|m_w\|$	$3\|G_1\|+\|m_w\|+\|m\|$	No
Wu et al. [22]	$\|G_1\|$	$2\|G_1\|+\|m_w\|$	$3\|G_1\|+\|m_w\|+\|m\|$	No
Ours	$2\|G_1\|$	$3\|G_1\|+\|m_w\|$	$6\|G_1\|+\|m_w\|+\|m\|$	Yes

7 Application

Mobile agent is a movable intelligent software program. It can implement a series of tasks according to the needs of users. Let's suppose a mobile agent who books flight tickets for a user in the Internet. In order to achieve unforgeability, the mobile agent must use a signature scheme. The Internet is an open environment overrun by hackers and viruses. If we use the user's long-term private key for signature operation, it has the risk of being exposed. A better strategy is to use a proxy signature. The user delegates his/her signing right to the mobile agent. Then the mobile agent acts as a proxy signer to sign messages on behalf of the user. The proxy private key is stored in the mobile agent instead of the long-term private key of the user. Even if the proxy private key in the mobile agent is exposed, it should not leak any information about the long-term private key. As our scheme is secure against proxy key exposure attack, it can be deployed in this scenario.

8 Conclusion

In this paper, we show that Hu et al.'s identity based proxy signature scheme in the standard model is insecure. We give four concrete kinds of attacks to their scheme. Then, we propose an improved scheme. We analysis the reasons why their scheme is insecure and the design principles of our improved scheme. We prove ours to be secure under the CDH assumption. At last, we evaluate the efficiency of our improved scheme, which shows that it is practical. The future work is to design identity-based proxy signature schemes which can resist the quantum attacks, such as schemes based on multivariate public key cryptography or lattice problems.

Acknowledgments

We would like to present our thanks to Ms. Yan Di, who checked our manuscript. This work is supported by the National Natural Science Foundation of China [Grant Nos. 61462048, 61362032, 61662039 and 61562047] and the Natural Science Foundation of Jiangxi Province, China (Grant No. 20151BAB207003 and 20161BAB202036).

References

[1] A. Shamir, Identity-based Cryptosystems and Signature Schemes, *Advances in Cryptology-CRYPTO 1984*, Santa Barbara, CA, 1984, pp. 47-53.

[2] D. Boneh, M. Franklin, Identity based Encryption from the Weil Pairing, *Advances in Cryptology- CRYPTO 2001*, Santa Barbara, CA, 2001, pp. 213-229.

[3] M. Mambo, K. Usuda, E. Okamoto, Proxy Signatures for Delegating Signing Operation, *Proceedings of the 3rd ACM Conference on Computer and Communications Security*, New Delhi, India, 1996, pp. 48-57.

[4] F. Zhang, K. Kim, Efficient Id-based Blind Signature and Proxy Signature from Bilinear Pairings, *Information Security and Privacy, 8th Australasian Conference, ACISP 2003*, Wollongong, Australia, 2003, pp. 312-323.

[5] J. Xu, Z. Zhang, D. Feng, ID-Based Proxy Signature using Bilinear Pairings, *Parallel and Distributed Processing and Applications - ISPA 2005 Workshops*, Nanjing, China, 2005, pp. 359-367.

[6] A. Boldyreva, A. Palacio, B. Warinschi, Secure Proxy Signature Schemes for Delegation of Signing Rights, *Journal of Cryptology*, Vol. 25, No. 1, pp. 57-115, January, 2012.

[7] X. Y. Huang, W. Susilo, Y. Mu, W. Wu, Proxy Signature Without Random Oracles, *Mobile Ad-hoc and Sensor Networks, Second International Conference, MSN 2006*, Hong Kong, China, 2006, pp. 473-484.

[8] D. Galindo, J. Herranz, E. Kiltz, On the Generic Construction of Identity-based Signatures with Additional Properties, *Advances in Cryptology-ASIACRYPT 2006*, Shanghai, China, 2006, pp. 178-193.

[9] F. Cao, Z. Cao, An Identity based Proxy Signature Scheme Secure in the Standard Model, *2010 IEEE International Conference on Granular Computing, GrC 2010*, San Jose, CA, 2010, pp. 67-72.

[10] Y. Sun, Y. Yu, X. S. Zhang, J. W. Chai, On the Security of An Identity-based Proxy Signature Scheme in the Standard Model, *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences*, Vol. E96-A, No. 3, pp. 721-723, March, 2013.

[11] K. Gu, W. Jia, C. Jiang, Efficient Identity-based Proxy Signature in the Standard Model, *Computer Journal*, Vol. 58, No. 4, pp. 792-807, April, 2015.

[12] D. B. He, M. W. Zhang, B. W. Xu, Insecurity of an Efficient Identity-based Proxy Signature in the Standard Model, *Computer Journal*, Vol. 58, No. 10, pp. 2507-2508, October, 2015.

[13] X. M. Hu, Y. C. Yang, J. Wang, H. J. Xu, W. N. Tan, Security Analysis of an Efficient Identity-based Proxy Signature in the Standard Model, *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences*, Vol. E98-A, No. 2, pp. 758-761, February, 2015.

[14] C. Gentry, Practical Identity-based Encryption Without Random Oracles, *Advances in Cryptology-EUROCRYPT 2006*, St. Petersburg, Russia, 2006, pp. 445-464.

[15] X. M. Hu, Y. C. Yang, Y. Liu, J. Wang, X. H. Xiong, A Highly Efficient and Identity-based Proxy Signature Scheme without Random Oracle, *2014 2nd International Conference on Information Technology and Electronic Commerce (ICITEC 2014)*, Dalian, China, 2014, pp 204-207.

[16] H. B. Tian, Z. T. Jiang, Y. Liu, B. D. Wei, A Systematic Method to Design Strong Designated Verifier Signature without Random Oracles, *Cluster Computing-the Journal of*

Networks Software Tools and Applications, Vol. 16, No. 4, pp. 817-827, December, 2013.

[17] X. M. Hu, H. Lu, H. J. Xu, J. Wang, Y. C. Yang, An Efficient Identity-based Proxy Signature Scheme in the Standard Model with Tight Reduction, *International Joint Conference – CISIS'15, 8th International Conference on Computational Intelligence in Security for Information Systems*, Burgos, Spain, 2015, pp. 309-319.

[18] X. M. Hu, J. Wang, H. J. Xu, Y. C. Yang, X. L. Xu, An Improved Efficient Identity-based Proxy Signature in the Standard Model, *International Journal of Computer Mathematics*, Vol. 94, No. 1, pp. 22-38, 2017.

[19] J. C. N. Schuldt, K. Matsuura, K. G. Paterson, Proxy Signatures Secure Against Proxy Key Exposure, *11th International Workshop on Practice and Theory in Public-Key Cryptography*, Barcelona, Spain, 2008, pp. 141-161.

[20] J. H. Liu, Q. H. Wu, J. W. Liu, T. Shang, Identity-based Proxy Multi-signature Applicable to Secure E-transaction Delegations, *High Technology Lettere*, Vol. 22, No. 2, pp. 199-206, June, 2016.

[21] K. A. Shim, An Identity-based Proxy Signature Scheme from Pairings, *Information and Communications Security, 8th International Conference, ICICS 2006*, Raleigh, NC, 2006, pp. 60-71.

[22] W. Wu, Y. Mu, W. Susilo, J. Seberry, X. Y. Huang, Identity-based Proxy Signature from Pairings, *Autonomic and Trusted Computing, 4th International Conference, ATC 2007*, Hong Kong, China, 2007, pp. 22-31.

[23] L. Chen, Z. Cheng, N. P. Smart, Identity-based Key Agreement Protocols from Pairings, *International Journal Information Security*, Vol. 6, No. 4, pp. 213-241, July, 2007.

Biographies

Caixue Zhou received the B.S. in Computer Science Department from Fudan University in 1988, Shanghai, China and the M.S. in Space College of Beijing University of Aeronautics and Astronautics in 1991, Beijing, China. Currently he is an associate professor with the School of Information Science and Technology, Jiujiang University, Jiujiang, China and a supervisor of postgraduate with the School of Information Technology, Jiangxi University of Finance and Economics, Nanchang, China. He is Member of the CCF (China Computer Federation) and Member of CACR (Chinese Association for Cryptologic Research). His research interests include applied cryptography and security of computer networks.

Zongmin Cui received the B.E degree from Southeast University in 2002 and the M.S. degree from HuaZhong University of Science and Technology in 2006. He received the Ph.D. Degree from Huazhong University of Science and Technology in 2014. He is currently an associate professor with the School of Information Science and Technology, Jiujiang University, Jiujiang, China. His research interests include cloud security, authorization update, key management, access control, and publish/subscribe system.

Guangyong Gao received the Ph.D. Degree from Nanjing University of Posts and Telecommunications, Nanjing, China, in 2012. Currently he is an associate professor with the School of Information Science and Technology, Jiujiang University, Jiujiang, China. His research interests include Multimedia Information Security, Digital Image Processing and Computer Networks Security.