J Inf Process Syst, Vol.12, No.2, pp.249~262, June 2016
http://dx.doi.org/10.3745/JIPS.03.0050

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Analysis of Warrant Attacks on Some Threshold Proxy Signature Schemes

Samaneh Mashhadi*

Abstract
In 2004, Yang et al. proposed a threshold proxy signature scheme that efficiently reduced the computational complexity of previous schemes. In 2009, Hu and Zhang presented some security leakages of Yang's scheme and proposed an improvement to eliminate the security leakages that had been pointed out. In this paper, we will point out that both Yang and Hu's schemes still have some security weaknesses, which cannot resist warrant attacks where an adversary can forge valid proxy signatures by changing the warrant m_W. We also propose two secure improvements for these schemes.

Keywords
Non-repudiation, Proxy Signature Scheme, Signature Scheme, Threshold Proxy Signature, Unforgeability

1. Introduction

The concept of a proxy signature was first introduced in [1]. A proxy signature scheme can be considered as a variation of the ordinary digital signature scheme [2], which enables a proxy signer to generate signatures on behalf of an original signer. So far, many proxy signature schemes have been discussed [1-15].

In a (t, n) threshold proxy signature scheme, the original signer conditionally delegates his/her authority of singing a message to a group of n members, the so-called proxy signers. The delegation condition is that any t or more proxy signers can corporately sign a message on behalf of the original signer, while any group of signers with less than t members cannot do so [1-15]. In general, a secure (t, n) threshold proxy signature scheme has the following inevitable properties: unforgeability, non-repudiation, secrecy, proxy protection, time constraint, and known signers. The unforgeability property is to ensure that any group of proxy signers with less than t members can never sign any message on behalf of the original signer. "Non-repudiation property" means that the proxy group cannot repudiate any proxy signature created by them, and the original signer cannot deny that he/she has delegated his/her authority of signing a message to the proxy group.

In 2004, Yang et al. [14] proposed a new threshold proxy signature scheme, which was more efficient than the previous one. In 2009, Hu and Zhang [5] presented frame and public-key substitute attacks on

Manuscript received January 28, 2015; first revision June 29, 2015; accepted August 28, 2015.
Corresponding Author: Samaneh Mashhadi (smashhadi@iust.ac.ir)
* Dept. of Mathematics, Iran University of Science & Technology, Tehran, Iran (smashhadi@iust.ac.ir)

Yang's scheme. This security leakage was treated in that paper. In this paper, we found the weakness of warrant in Yang and Hu's schemes. A warrant attack is when a malicious original signer or any proxy signer can forge valid proxy signatures by changing the warrant m_W. So far, various kinds ofwarrant attacks on proxy signature schemes have been discussed [3,7,11,13]. In this paper, we point out that both Yang and Hu's schemes still have some security weaknesses, which cannot resist warrant attacks. To remedy these weaknesses, we propose two new improvements for these schemes with higher safety. The rest of this paper is laid out as follows: In Section 2, we review Yang's scheme. In Section 3, we point out the security leak inherent in Yang's scheme. In Sections 4 and 5 we review Hu's scheme and show that it cannot resist a warrant attack, and we propose a method to eliminate this weakness. A novel improvement for Yang's scheme is proposed in Section 6. In Section 7, we analyze the security of our scheme. The performance of the proposed scheme is discussed in Section 8. Finally, we give our conclusions in Section 9.

2. Brief Review of Yang's Scheme

In this section, we briefly review Yang's scheme [14]. The scheme consists of 4 phases: initialization, proxy share generation, proxy signature generation, and proxy signature verification.

2.1 Initialization Phase

Let p be a large prime, q be a prime divisor of $p-1$, g be a generator of order q in $\mathbb{Z}_p^*$, and $h(.)$ be a secure one-way hash function. The parameters (p, q, g) are public. Suppose that P_0 stands for the original signer, and let $G = \{P_1, P_2, \dots, P_n\}$ be the proxy group of n proxy signers. The original signer P_0 determines its private key by choosing an arbitrary $x_0 \in \mathbb{Z}_q^*$ and the public key as $y_0 = g^{x_0} mod p$. By the same way, each proxy signer $P_i \in G$ owns its private key $x_i \in \mathbb{Z}_q^*$ and public key $y_i = g^{x_i} mod p$, which are certified by the certificate authority (CA). Let m_W stand for a warrant that records the parameters t, n, the valid delegation time, and the identities of the original and proxy signers of the proxy group, etc. Also, $ASID$ denotes the identities of the actual proxy signers.

2.2 Proxy Share Generation Phase

The original signer P_0 chooses a random integer $k \in \mathbb{Z}_q^*$ and then computes $K = g^k mod p$. Then, $\boldsymbol{P_0}$ computes $\sigma = x_0 h(m_W, K) + k mod q$ as the proxy group's key and reports (σ, m_W, K) to the proxy signers of G. After receiving (σ, m_W, K), each proxy signer $P_i \in G$ checks whether the equation $g^\sigma = y_0^{h(m_W, K)} K mod p$ holds or not. If it holds, each P_i regards σ as its proxy key.

2.3 Proxy Signature Generation Phase

For convenience, let $D = \{P_1, P_2, \dots, P_t\}$ be the t actual proxy signers, $ASID$ be the identities of them, C be the receiver, and m be the message to be signed. Then, D as a proxy group performs the following steps: 1) each $P_i \in D$ chooses a random integer $k_i \in \mathbb{Z}_q^*$, and then computes and reports $r_i = g^{k_i} mod p$; 2) after receiving r_j $(j = 1, 2, \dots, t, j \neq i)$, each $P_i \in D$ computes $R = \prod_{j=1}^{t} r_j \, mod p$ and then

$S_i = k_i R + (t^{-1}\sigma + x_i) h(R, m, ASID) mod q$; 3) they send S_i to the designated receiver C via a secret channel; and 4) after receiving S_i, the receiver C checks whether the following equation holds:

$$g^{S_i} = r_i^R \left[\left(K y_0^{h(m_W, K)} \right)^{t^{-1}} y_i \right]^{h(R,m,ASID)} mod p \, . \tag{1}$$

If it holds, (r_i, S_i) is a valid partial proxy signature; then he computes $S = \sum_{i=1}^{t} S_i mod q$. Therefore, $(R, S, K, m_W, ASID)$ is the threshold proxy signature of the message m.

2.4 Proxy Signature Verification Phase

From m_W, the verifier can get the threshold value t, and from $ASID$, he can know the number of actual proxy signers. The verifier checks the validity of the proxy signature $(R, S, K, m_W, ASID)$ for the message m by checking the validity of the following equation

$$g^S = R^R \left(K y_o^{h(m_W, K)} \prod_{i=1}^{t} y_i \right)^{h(R,m,ASID)} mod p. \tag{2}$$

3. Our Attacks on Yang's Scheme

In this section, we show that Yang's scheme cannot resist warrant attacks (i.e., the warrant m_W in the proxy signature can be replaced by any other warrant the adversary wants). Therefore, the adversary can forge a valid proxy signature.

3.1 Case 1

In this case, we describe that after intercepting a valid proxy signature $(R, S, K, m_W, ASID)$, an adversary (each malicious original or proxy signer) can change m_W, y_i and forge a proxy signature S (without knowing or changing secret partial signature S_j).

Without loss of generality, assume that P_1 is a malicious proxy signer who decides to forge a threshold proxy signature of a message m. Although, P_1 cannot generate a valid warrant m_W of the original signer, he can generate a valid warrant m'_W. As a result, he can change the content of the warrant such as the threshold value t, the time constraint, etc.

Assume that P_1 decides to forge a threshold proxy signature of m and claim that it is generated by t' proxy signers $D' = \left\{P_1, P_2, \dots, P_{t'}\right\}$, while the proxy group D' knows nothing about the decision. Let $ASID'$ be the identities of the group D'. First, P_1 generates the warrant m'_W as he wants, chooses two random integers $\alpha, \beta \in \mathbb{Z}_q^*$, and computes:

$$R' = g^{\beta} mod p, \; K' = g^{\alpha} mod p. \tag{3}$$

Then, he computes

$$y_1 = y_0^{-h(m'_W,K')}\left(\prod_{i=2}^{t'} y_i\right)^{-1} modp, \tag{4}$$

and requests CA to replace his public key by y_1. Next he computes:

$$S' = \beta R' + \alpha h(R', m, ASID') modp. \tag{5}$$

Since:

$$g^{S'} = g^{\beta R'} g^{\alpha h(R',m,ASID')} modp = R'^{R'} g^{\alpha h(R',m,ASID')} = R'^{R'} \left(K' y_o^{h(m'_W,K')} \prod_{i=1}^{t'} y_i\right)^{(R',m,ASID')} \tag{6}$$

$(R', S', K', m'_W, ASID')$ is a valid threshold proxy signature of the message m. This kind of attack can be prevented using a policy such as restricting the proxy to update their keys or if CA asks P_1 for the Zero-Knowledge Proof of his private key x'_1 associated to new public key y_1.

3.2 Case 2

In 2007, Shao et al. [11], proposed another warrant attack on Yang's scheme. In this case, we review this attack. Shao described how after intercepting a valid proxy signature $(R, S, K, m_W, ASID)$ an adversary (each malicious original or proxy signer) can change m_W, σ and forge a proxy signature S (without knowing or changing secret partial signature S_j) in Yang's scheme.

Suppose that a malicious original signer P_0 decides to forge a proxy signature generated by the proxy group $D = \{P_1, P_2, \dots, P_t\}$. Let $ASID$ be the identities of D. Let $(R, S, K, m_W, ASID)$ be a legal proxy signature of a message m generated by D on behalf of P_0. The malicious original signer P_0 can change the content of the warrant, such as the time constraint. P_0 forges a new warrant m'_W as he wants, chooses an integer $k' \in \mathbb{Z}_q^*$ and computes $K' = g^{k'} modp$. Then, P_0 computes $\sigma' = x_0 h(m'_W, K') + k' modq$ and:

$$S' = S + (\sigma' - \sigma) h(R, m, ASID) modq. \tag{7}$$

Then, $(R, S', K', m'_W, ASID)$ can pass the verification equation, since:

$$g^{S'} = g^S g^{(\sigma'-\sigma)h(R,m,ASID)} = g^{\sum_{i=1}^{t} S_i} g^{(\sigma'-\sigma)h(R,m,ASID)}$$

$$= g^{\sum_{i=1}^{t}(k_i R + x_i h(R,m,ASID))} g^{\sigma' h(R,m,ASID)}$$

$$= R^R \left(K' y_o^{h(m'_W,K')} \prod_{i=1}^{t} y_i\right)^{h(R,m,ASID)} modp. \tag{8}$$

4. Brief Review of Hu's Scheme

In this section, we review Hu's scheme [5]. The scheme consists of 4 phases: initialization, proxy share generation, proxy signature generation, and proxy signature verification.

4.1 Initialization Phase

The system parameters are the same as those in Section 2.1. However, the only difference is that in Hu's scheme, CA requires that the original signer P_0 and each proxy signer $P_i (1 \le i \le n)$ offer the Zero-Knowledge Proof of its private key associated with its public key as follows: 1) CA randomly chooses $e \in \mathbb{Z}_q^*$, computes $E = g^e mod p$, and sends E to P_0 to each P_i; 2) then, P_i, for $i = 0,1,\dots,n$, computes $L_i = E^{x_i} mod p$, and sends L_i to CA; and 3) for each $i = 0,1,\dots,n$, the certificate authority (CA) checks the equation $y_i^e = L_i$; if it holds, CA accepts their certification, otherwise he refuses it.

4.2 Proxy Share Generation Phase

The original signer P_0 chooses a random integer $k \in \mathbb{Z}_q^*$ and then computes $K = g^k mod p$. Then, P_0 computes $\sigma = x_0 h(m_W, K) + kK mod q$ as the proxy group's key. Accordingly, its public key is $Q = g^\sigma mod p$. Then, P_0 chooses a $t-1$ degree polynomial $f(x) = \sigma + a_1 x + \cdots + a_{t-1} x^{t-1} mod p$ and computes $R_i = f(y_i) mod p$ as each proxy signer secret key. He computes $Q_i = g^{R_i} mod p, A_j = g^{a_j} mod p$ and sends (y_i, R_i, m_W, K) to each proxy signer P_i via a secret channel and broadcasts Q_i, A_j. After receiving (y_i, R_i, m_W, K), each proxy signer $P_i \in G$ checks whether the equation $g^{R_i} = K^K y_o^{h(m_W,K)} \prod_{l=1}^{t-1} A_l^{y_i^l} mod p$ holds or not.

4.3 Proxy Signature Generation Phase

For convenience, let $D = \{P_1, P_2, \dots, P_t\}$ be the t actual proxy signers, $ASID$ be the identities of them, C be the receiver, and m be the message to be signed. Then, D as a proxy group performs the following steps: 1) each $P_i \in D$ chooses a random integer $k_i \in \mathbb{Z}_q^*$ and then computes and reports $r_i = g^{k_i} mod p$; 2) after receiving r_j $(j = 1, 2, \dots, t, j \ne i)$, each $P_i \in D$ computes $R = \prod_{j=1}^t r_j mod p$ and the $S_i = k_i R + (R_i W_i + x_i) h(R, m, ASID) mod q$, where $W_i = \prod_{j=1, j\ne i}^t {}^{y_j}/_{y_j - y_i}$; 3) each P_i sends S_i to the designated receiver C via a secret channel; and 4) after receiving S_i, the receiver C checks whether the following equation holds:

$$g^{S_i} = r_i^R \left[Q_i^{W_i} y_i\right]^{h(R,m,ASID)} mod p \tag{9}$$

If it holds, C computes $S = \sum_{i=1}^t S_i mod q$. Therefore, $(R, S, K, m_W, ASID)$ is the threshold proxy signature of the message m.

4.4 Proxy Signature Verification Phase

The verifier checks the validity of the proxy signature $(R, S, K, m_W, ASID)$ for the message m by checking the validity of the following equation:

$$g^S = R^R \left(K^K y_o^{h(m_W,K)} \prod_{i=1}^t y_i\right)^{h(R,m,ASID)} modp. \tag{10}$$

5. Warrant Attack on Hu's Scheme

5.1 Warrant Attack

In the following section, we show that Hu's scheme cannot resist warrant attacks, similarly to how we did so in Subsection 3.2. After intercepting a valid proxy signature generated by a subset of a proxy group, a malicious original signer can change the warrant and forge new proxy signatures.

Suppose that a malicious original signer P_0 wants to forge proxy signatures generated by the proxy group $D = \{P_1, P_2, \dots, P_t\}$. Let $ASID$ be the identities of D. Let $(R, S, K, m_W, ASID)$ be a legal proxy signature of a message m generated by D on behalf of P_0. The malicious original signer P_0 can change the content of the warrant, such as the time constraint, etc. P_0 forges a new warrant m'_W as he wants, chooses an integer $k' \in \mathbb{Z}_q^*$, and computes $K' = g^{k'} modp$. Then, P_0 computes $\sigma' = x_0 h(m'_W, K') + k'K' modq$ and replaces $Q = g^\sigma modp$ with $Q' = g^{\sigma'} modp, Q_i = g^{R_i} modp$ for $1 \le i \le n$, and A_j for an arbitrary $1 \le j \le t, j \ne i$ with arbitrary numbers Q'_i for $1 \le i \le n$ and A'_j, respectively. Then, P_0 computes:

$$S' = S + (\sigma' - \sigma)h(R, m, ASID)modq. \tag{11}$$

Similarly, $(R, S', K', m'_W, ASID)$ can pass the verification equation, since:

$$g^{S'} = g^S g^{(\sigma'-\sigma)h(R,m,ASID)} = g^{\sum_{i=1}^t S_i} g^{(\sigma'-\sigma)h(R,m,ASID)} = g^{\sum_{i=1}^t k_i R + (R_i W_i + x_i)h(R,m,ASID)} g^{(\sigma'-\sigma)h(R,m,ASID)}$$

$$= R^R (g^\sigma \prod_{i=1}^t y_i)^{h(R,m,ASID)} g^{(\sigma'-\sigma)h(R,m,ASID)} = R^R \left(g^{\sigma'} \prod_{i=1}^t y_i\right)^{h(R,m,ASID)}$$

$$= R^R \left({K'}^{K'} y_o^{h(m'_W,K')} \prod_{i=1}^t y_i\right)^{h(R,m,ASID)} modp. \tag{12}$$

5.2 Our Improvement

In order to protect Hu's scheme against the above attack, we recommend that the partial signature S_i be replaced by:

$$S_i = k_i R + (R_i W_i + x_i)h(R, m, m_W, ASID), \tag{13}$$

the partial signature verification equation is replaced by:

$$g^{S_i} = r_i^R \left[Q_i^{W_i} y_i\right]^{h(R,m,m_W,ASID)} modp, \tag{14}$$

and the threshold proxy signature verification equation is replaced by:

$$g^S = R^R \left(K^K y_o^{h(m_W,K)} \prod_{i=1}^{t} y_i\right)^{h(R,m,m_W,ASID)} \tag{15}$$

Since m_W is a part of S_i, it is impossible for anyone to change m_W and forge a proxy signature after intercepting a valid proxy signature $(R, S, K, m_W, ASID)$. Thus, a malicious original signer cannot forge a valid threshold proxy signature by a warrant attack.

6. A Novel Proxy Signature Scheme

In this section, we propose an improvement to the Yang scheme. Our scheme can be divided into 4 phases: initialization, proxy share generation, proxy signature generation, and proxy signature verification.

6.1 Initialization Phase

This phase is similar to Hu's scheme, CA requires that the original signer P_0 and each proxy signer $P_i (1 \leq i \leq n)$ offer the Zero-Knowledge Proof of its private key associated with its public key as follows: CA randomly chooses $a \in \mathbb{Z}_q^*$, computes $A = g^a mod p$, and sends A to P_0 to each P_i. Then, P_i for $i = 0,1, \dots, n$ computes $A_i = A^{x_i} mod p$ and sends A_i to CA. For each $i = 0,1, \dots, n$, the CA checks the equation $y_i^a = A_i$. If it holds, CA accepts their certification, otherwise he refuses it.

6.2 Proxy Share Generation Phase

This phase is the same as that in Subsection 2.2.

6.3 Proxy Signature Generation Phase

For convenience, let $D = \{P_1, P_2, \dots, P_t\}$ be t actual proxy signers, $ASID$ the identities of these t proxy signers, C the receiver of partial signatures, and m a message to be signed. Then D, as a proxy group, performs the following steps: 1) each $P_i \in D$ chooses a random $k_i \in \mathbb{Z}_q^*$ and then computes and broadcasts $r_i = g^{k_i} mod p$; 2) after receiving r_j $(j = 1, 2, \dots, t, j \neq i)$, each $P_i \in D$ computes:

$$R = \prod_{j=1}^{t} r_j \, mod p, \tag{16}$$

$$S_i = k_i + (t^{-1}\sigma + x_i K) h(R, m, m_W, ASID) mod q. \tag{17}$$

3) then, he sends S_i to the designated receiver C via a secret channel; and 4) after receiving S_i, the receiver C checks whether the following equation holds:

$$g^{S_i} = r_i \left[\left(K y_0^{h(m_W,K)}\right)^{t^{-1}} y_i^K\right]^{h(R,m,m_W,ASID)} mod p. \tag{18}$$

If it holds, (r_i, S_i) is a valid partial proxy signature, and then he computes $S = \sum_{i=1}^{t} S_i$. Therefore, $(R, S, K, m_W, ASID)$ is the threshold proxy signature of the message m.

6.4 Proxy Signature Verification Phase

The verifier checks the validity of the proxy signature $(R, S, K, m_W, ASID)$ for the message m by the following equation:

$$g^S = R\left[Ky_o^{h(m_W,K)}(\textstyle\prod_{i=1}^{t} y_i)^K\right]^{h(R,m,m_W,ASID)} modp. \tag{19}$$

7. Security Analysis

In the following section, we first proof some lemma and theorems and then examine the security of the proposed scheme (subsections 7.1–7.6).

Lemma 1. If $\prod_{i=1}^{t} y_i = g^c \bmod p$, and $g^r = K' = (\prod_{i=1}^{t} y_i)^{-K'} g^\alpha \bmod p$, then $K' = (\alpha - r)c^{-1} \bmod q$.

Proof. Indeed, we have:

$$\begin{aligned} g^r = K' &= (\textstyle\prod_{i=1}^{t} y_i)^{-K'} g^\alpha \bmod p \\ &= g^{-cK'} g^\alpha \bmod p \\ &= g^{-cK'+\alpha} \bmod p. \end{aligned} \tag{20}$$

Thus, $K' = (\alpha - r)c^{-1} \bmod q$.

Theorem 2. $(R', S', K', m_W, ASID)$ given by:

$$\begin{aligned} &R = g^\beta \bmod p, \\ &K' = (\textstyle\prod_{i=1}^{t} y_i)^{-K'} g^\alpha \bmod p, \quad and \\ &S' = \beta + [\alpha + x_0 h(m_w, K')]h(R', m, m_W, ASID). \end{aligned} \tag{21}$$

is a valid proxy signature.

Proof. It is a valid proxy signature of the message m because:

$$\begin{aligned} g^{S'} &= g^\beta (g^{\alpha + x_0 h(m_w,K')})^{h(R',m,m_W,ASID)} \\ &= R'(K' y_0{}^{h(m_w,K')}(\textstyle\prod_{i=1}^{t} y_i)^{K'})^{h(R',m,m_W,ASID)}. \end{aligned} \tag{22}$$

Theorem 3. If $y_1 = (y_0{}^{h(m_w,K')}(\prod_{i=2}^{t} y_i)^{K'})^{-K'^{-1}} \bmod p$, then $(R', S', K', m_W, ASID)$ given by:

$$\begin{aligned} &R' = g^\beta \bmod p, \quad K' = g^\alpha \bmod p, \quad and \\ &S' = \beta + \alpha h(R', m', m_W, ASID) \bmod q, \end{aligned} \tag{23}$$

is a valid proxy signature.

Proof. It is a valid proxy signature of the message $m^{'}$ because:

$$\begin{aligned} g^{S^{'}} &= g^{\beta} g^{\alpha h(R^{'},m^{'},m_W,ASID)} \\ &= g^{\beta}(g^{\alpha} y_0^{h(m_W,K^{'})} y_0^{-h(m_W,K^{'})} (\textstyle\prod_{i=2}^{t} y_i)^{K^{'}} \\ &\quad (\textstyle\prod_{i=2}^{t} y_i)^{-K^{'}})^{h(R^{'},m^{'},m_W,ASID)} \\ &= R'(K' {y_0}^{h(m_W,K^{'})} (\textstyle\prod_{i=1}^{t} y_i)^{K^{'}})^{h(R^{'},m^{'},m_W,ASID)}. \end{aligned} \tag{24}$$

Next, we examine the security of the proposed scheme.

7.1 Secrecy

In the proposed scheme, both signing and verification are based on discrete logarithm problems. Hence, no one can compute the original signer's private key x_0 from his/her public key $y_0 = g^{x_0} \bmod p$. Similarly, no one can compute the original signer's private key x_0 from $A_0 = A^{x_0} \bmod p$. On the other hand, no one can compute x_0 from the group proxy signature key $\sigma = x_0 h(m_W, K) + k \bmod q$, because the parameter σ is computed by the Schnorr signature scheme, which is a provable secure random oracle model. Therefore, the original signature's private key can be kept secretly and be reused during the span of the system.

Based on discrete logarithms, it is virtually impossible to obtain any proxy signer's secret key x_i from the corresponding public key $y_i = g^{x_i} \bmod p$ or from $A_i = A^{x_i} \bmod p$. Again, according to the Schnorr signature scheme, no one can compute x_i from the partial proxy signature $S_i = k_i + (t^{-1}\sigma + x_i K)h(R, m, m_W, ASID)$. Hence, our scheme preserves the security.

7.2 Proxy Protection

Although the proxy signing key σ is created by the original signer, the original signer cannot compute the partial proxy signature,

$$S_i = k_i + (t^{-1}\sigma + x_i K)h(R, m, m_W, ASID). \tag{25}$$

The original signer does not know P_i's private key x_i, so according to the Schnorr signature scheme, it is very difficult for anyone to generate the partial proxy signature S_i of m. For the security of the Schnorr signature scheme, the random number k_i should not be reused with a different plain text. Therefore, the original signer cannot masquerade as a proxy signer to create a partial proxy signature. This protects the authority of the proxy signer.

7.3 Unforgeability

An intruder may try to derive a forged proxy signature by various ways. In the subsections below we will show that our scheme is secure against various attacks.

Attack 1: An attacker may try to derive P_i's private key x_i from S_i.

- Analysis: Once P_i broadcasts S_i, the intruder cannot derive x_i from S_i because the random number k_i is unknown.

Attack 2: An attacker may try to forge P_i's partial signature S_i.

- Analysis: Without the proxy signing key σ and P_i's private key x_i, no one can forge the proxy signer P_i to construct $S_i = k_i + (t^{-1}\sigma + x_i K)h(R, m, m_W, ASID)$ mod$q$.

Frame attack: A malicious original signer P_0, without any knowledge about P_i's private key x_i, may try to forge a valid general proxy signature $(R', S', K', m_W, ASID)$ for his/her arbitrary chosen message m' and dishonestly claim that it is generated by other t proxy signers $D = \{P_1, P_2, \dots, P_t\}$.

- Analysis: Let $ASID$ be the identities of D. For this purpose, P_0 can choose random integers $\alpha, \beta \in \mathbb{Z}_q^*$ and compute $R' = g^\beta \bmod p$. Now, according to Theorem 2, P_0 should determine:

$$K' = \left(\prod_{i=1}^{t} y_i\right)^{-K'} g^{\alpha} mod p, \tag{26}$$

and:

$$S' = \beta + [\alpha + x_0 h(m_W, K')]h(R', m, m_W, ASID). \tag{27}$$

However, according to Lemma 1, P_0 should solve the discrete logarithm problem $\prod_{i=1}^{t} y_i = g^c \bmod p$ in order to compute K'. Thus, P_0 cannot forge a valid general proxy signature of any message m' that is generated by D.

Public key substitute attack: Without the loss of generality, suppose that a malicious proxy signer P_1 decides to forge a general proxy signature scheme of a message m' by himself or herself, without the assistance of other proxy signers.

- Analysis: For this purpose, according to Theorem 3, P_1 chooses random $\alpha, \beta \in \mathbb{Z}_q^*$ and computes

$$\begin{aligned} &R' = g^\beta \bmod p, \quad K' = g^\alpha \bmod p, \\ &S' = \beta + \alpha h(R', m', m_W, ASID) \bmod q, \\ &y_1 = \left({y_0}^{h(m_W, K')}\left(\prod_{i=2}^{t} y_i\right)^{K'}\right)^{-K'^{-1}} \bmod p. \end{aligned} \tag{27}$$

Then he wants CA to replace his public key with the above y_1. The certificate authority, CA, again asks P_1 for the Zero-Knowledge Proof of his private key x'_1 associated to new public key y_1, but P_1 cannot obtain x'_1, s.t. $y_1 = g^{x'_1} \bmod p$ because of the difficulty of solving the discrete logarithm problem. Hence, P_1 cannot again perform a Zero-Knowledge Proof with CA when he changes his public key.

Warrant attack: After intercepting a valid proxy signature $(R, S, K, m_W, ASID)$ the adversary may try to replace (m_W, σ) with (m'_W, σ').

- Analysis: However, m_W is protected under the hash function, $h(R, m, m_W, ASID)$ in the individual signature $S_i = k_i + (t^{-1}\sigma + x_i K)h(R, m, m_W, ASID)$mod q. So, the probability of obtaining m'_W such that $h(R, m, m_W, ASID) = h(R, m, m_{W'}, ASID)$ is equivalent to performing an exhaustive search on m'_W. Thus, after intercepting a valid proxy signature $(R, S, K, m_W, ASID)$, it is impossible for anyone to change (m_W, σ) Hence, our scheme can resist Shao's warrant attack.

Warrant attack: An attacker may generate a warrant $m_{W'}$ and try to forge a threshold proxy signature of a message m'.

- Analysis: For this purpose, similar to Theorem 2, P_1 chooses random $\alpha, \beta \in \mathbb{Z}_q^*$ and computes:

$$
\begin{aligned}
&R' = g^{\beta} \bmod p, \quad K' = g^{\alpha} \bmod p, \\
&S' = \beta + \alpha h(R', m', m_{W'}, ASID') \bmod q, \\
&y_1 = ({y_0}^{h(m_{W'}, K')}(\textstyle\prod_{i=2}^{t} y_i)^{K'})^{-K'^{-1}} \bmod p.
\end{aligned}
\tag{29}
$$

Then, he wants CA to replace his public key with the above y_1. The certificate authority, CA, again asks P_1 for the Zero-Knowledge Proof of his private key x'_1 associated with the new public key y_1. However, P_1 cannot obtain x'_1, s.t. $y_1 = g^{x'_1} \bmod p$ because of the difficulty of solving the discrete logarithm problem. Hence, P_1 cannot again perform the Zero-Knowledge Proof with CA when he changes his public key.

Similarly, no proxy signer P_i can forge a valid threshold proxy signature with this type of attack.

Attack 7: $t - 1$ or fewer proxy signers may try to sign a message m.

- Analysis: The attackers may try to derive a forged proxy signature by using the previous attacks. But, we have shown that all attacks fail on our scheme. The proxy signature can be only generated by any t or more delegated proxy signers. Since the threshold value t is defined in the warrant m_W, if the number of actual proxy signers does not achieve t, the proxy signature is invalid. Furthermore, as discussed above, we know that our improved scheme can resist warrant attacks. Therefore, our scheme satisfies the property of *unforgeability*.

7.4 Non-repudiation

The property of *non-repudiation* is that both the original signer and the actual proxy signers cannot deny the generation of a valid proxy signature. Any valid proxy signature $(R, S, K, m_W, ASID)$ of a message m should be generated by t or more proxy signers. This is because only P_i has the private key x_i. Thus, P_i cannot deny signing the partial proxy signature. Moreover, the warrant m_W and K are created by the original signer. The original signer cannot deny the proxy signers the power of signing messages. Therefore, the valid proxy signature can be signed on behalf of the original signer. Hence, both the original signer and the actual proxy signers cannot deny generating the valid proxy signature.

7.5 Time Constraint

Time constraint means the time during which the signing power of the proxy group is valid. In our scheme, only the original signer creates the warrant m_W, which contains the time constraint, and it is impossible for anyone to change m_W. In the verification stage, the verifier checks whether or not the warrant has expired. Therefore, our scheme satisfies the property of time constraint.

7.6 Known Signers

Finally, from $ASID$, the verifier can notice who the actual signers are. In our scheme, any receiver is able to identify the actual signers in the proxy group. Furthermore, the adversary cannot replace $ASID$ by $ASID'$ satisfying $h(R, m, m_W, ASID) = h(R, m, m_W, ASID')$. Since $h(\cdot)$ is a collision resistant hash function, it is computationally infeasible to get such an $ASID'$. Therefore, our scheme satisfies the property of *known signers*.

From what has been analyzed above, we are certain that the necessary requirements of the (t, n) threshold proxy signature scheme are fulfilled in our scheme. Moreover, we compared the security of our scheme with the threshold proxy signature schemes proposed in [5,11,14] and summarized the results in Table 1.

Table 1. Security comparison of threshold schemes with proposed scheme

Security features	Yang	Shao	Hu	Our scheme
Secrecy	Yes	Yes	Yes	Yes
Proxy protection	Yes	Yes	Yes	Yes
Unforgeability	No	No	Yes	Yes
Non-repudiation	No	No	Yes	Yes
Time constraint	No	Yes	Yes	Yes
Known signers	No	No	Yes	Yes
Secure channel	No	No	No	No
Scheme can resist frame attacks	No	No	Yes	Yes
Scheme can resist public-key substitute attacks	No	No	Yes	Yes
Scheme can resist warrant attacks	No	Yes	Yes	Yes
At least t proxy signers can generate valid proxy signature	No	No	Yes	Yes

8. Performance

In this section, we compare the complexity of the new proxy signature scheme with that of the threshold proxy signature schemes proposed in [4,5,11,12,14]. The results are summarized in Tables 2 and 3. For convenience, the following notations were used to analyze computational complexity.

T_e the time for one exponentiation computation.

T_m the time for one mod ular multiplication computation.

T_H the time for hash function computation.

T_i the time for one inverse computation.

The time complexities for modular exponentiation, multiplication, and inverse computation are $O(log^3 p)$, $O(log^2 p)$, and $O(log^2 p)$, respectively. As shown in Table 2, the computational complexity of our presented scheme for share generation, signature generation, and signature verification are $3T_e + 2T_m + T_H$, $(3t + 2)T_e + (2t + 3)T_m + 2T_H + T_i$, and $3T_e + (t + 2)T_m + 2T_H$, respectively, which are less than those of previous schemes. Also, from Table 3, we can see that the overall computation costs of our improved scheme is $(3t + 8)T_e + (3t + 7)T_m + 5T_H + T_i$, which is less than that of previous schemes.

Therefore, our scheme can reduce computation costs, and it is the most efficient and the most secure non-repudiable threshold proxy signature scheme with known signers.

Also, a comparison of our attacks with the previous attacks on Yang's scheme is given in Table 4.

Table 2. Computational complexities

Scheme	Share generation	Signature generation	Signature verification
Sun [12]	$(5n+2t)T_e+(nt+2t)T_m+T_H$	$(4t^2-t)T_e+(10t^2-14t)T_m+2T_H+(t^2-t)T_i$	$4T_e+tT_m+2T_H$
Hsu et al. [4]	$(5n+2t)T_e+(nt+2t)T_m+T_H$	$(t^2+4t)T_e+(4t^2+2t)T_m+2T_H+t^2T_i$	$4T_e+tT_m+2T_H$
Yang et al. [14]	$3T_e+2T_m+T_H$	$4tT_e+3tT_m+2T_H+T_i$	$4T_e+tT_m+2T_H$
Shao et al. [11]	$3T_e+2T_m+T_H$	$4tT_e+3tT_m+2T_H$	$4T_e+tT_m+2T_H$
Hu and Zhang [5]	$(4n+t)T_e+ntT_m+T_H$	$4tT_e+3tT_m+2T_H$	$5T_e+tT_m+2T_H$
Present study	$3T_e+2T_m+T_H$	$3tT_e+2tT_m+2T_H+T_i$	$3T_e+tT_m+2T_H$

Table 3. Overall computational complexities

Scheme	Overall
Sun [12]	$4t^2T_e+(10t^2+(n-11)t)T_m+5T_H+t^2)T_i$
Hsu et al. [4]	$(4t^2+5n)T_e+(4t^2+(n+5)t)T_m+5T_H+(t^2-1)T_i$
Yang et al. [14]	$(4t+9)T_e+(4t+7)T_m+5T_H+T_i$
Shao et al. [11]	$(4t+9)T_e+(4t+7)T_m+5T_H$
Hu and Zhang [5]	$(4n+5t)T_e+((n+4)t)T_m+5T_H$
Present study	$(3t+8)T_e+(3t+7)T_m+5T_H+T_i$

Table 4. Comparison of attacks

Attack on	Attack	Method of attack
Yang scheme	Frame attack [13]	$m \to m'$, but y_i, m_W are constant
Yang scheme	Warrant attack [11]	$m_W \to m'_W$, but y_i, m are constant
Yang scheme	Public-key attack [5]	$m \to m'$, $y_i \to y'_i$ but m_W is constant
Yang scheme	Warrant attack (present study)	$m_W \to m'_W$, $y_i \to y'_i$, but m is constant
Hu scheme	Warrant attack (present study)	$m_W \to m'_W$, but y_i, m are constant

9. Conclusion

In this paper, we have pointed out the both Yang and Hu's schemes still have some security weaknesses, which cannot resist warrant attacks. Finally, to remedy these weaknesses, we proposed new improvements for these schemes.

References

[1] M. Mambo, K. Usuda, and E. Okamoto, "Proxy signature: delegation of the power to sign messages," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 79A, no. 9, pp. 1338-1354, 1996.

[2] S. Huang and C. H. Shi, “A simple multi-proxy signature scheme,” in *Proceedings of the 10th National Conference on Information Security*, Hualien, Taiwan, 2000, pp. 134-138.

[3] H. Bao, Z. Cao and S. Wang, “Improvement on Tzeng et al.’s nonrepudiable threshold multi-proxy multi-signature scheme with shared verification,” *Applied Mathematics and Computation*, vol. 169, no. 2, pp. 1419-1430, 2005.

[4] C. Hsu, T. Wu, and T. Wu, “New nonrepudiable threshold proxy signature scheme with known signers,” *Journal of Systems and Software*, vol. 58, no. 2, pp. 119-124, 2001.

[5] J. Hu and J. Zhang, “Cryptanalysis and improvement of a threshold proxy signature scheme,” *Computer Standards and Interfaces*, vol. 31, no. 1, pp. 169-173, 2009.

[6] H. F. Huang and C. C. Chang, “A novel efficient (t,n) threshold proxy signature scheme,” *Information Sciences*, vol. 176, no. 10, pp. 1338-1349, 2006.

[7] S. Mashhadi, “Analysis of frame attack on Hsu et al.’s non-repudiable threshold multi-proxy multi-signature scheme with shared verification,” *Scientia Iranica*, vol. 19, no. 3, pp. 674-679, 2012.

[8] S. Mashhadi, “A novel non-repudiable threshold proxy signature scheme with known signers,” *International Journal of Network Security*, vol. 15, no. 4, pp. 231-236, 2013.

[9] S. Mashhadi and M. Abdi, “A Secure non-repudiable general proxy signature,” *International Journal of Cyber-Security and Digital Forensics*, vol. 4, no. 2, pp. 380-389, 2015.

[10] S. Mashhadi, “A novel secure self proxy signature scheme,” *International Journal of Network Security*, vol. 14, no. 1, pp. 22-26, 2012.

[11] J. Shao, Z. Cao, and R. Lu, “Improvement of Yang et al.’s threshold proxy signature scheme,” *Journal of Systems and Software*, vol. 80, no. 2, pp. 172-177, 2007.

[12] H. M. Sun, “An efficient nonrepudiable threshold proxy signature scheme with known signers,” *Computer Communications*, vol. 22, no. 8, pp. 717-722, 1999.

[13] Z. Tan, Z. Liu, and M. Wang, “On the security of some nonrepudiable threshold proxy signature schemes,” in *Information Security Practice and Experience*. Heidelberg: Springer, 2005, pp. 374-385.

[14] C. Y. Yang, S. F. Tzeng, and M. S. Hwang, “On the efficiency of nonrepudiable threshold proxy signature scheme with known signers,” *Journal of Systems and Software*, vol. 73, no. 3, pp. 507-514, 2004.

[15] K. Zhang, “Threshold proxy signature schemes,” in *Information Security.* Heidelberg: Springer, 1997, pp. 191-197.

Samaneh Mashhadi

She was born in Tafresh, Iran, on March 27, 1982. She received the B.Sc. and M.Sc. degrees with honors in Mathematics from Iran University of Science and Technology (IUST), and Amirkabir University of Technology (AUT) in 2003 and 2005, respectively. She received her Ph.D. with honors in Mathematics (Cryptography) in 2008 from IUST. She is currently an Assistant Professor in Department of Mathematics of IUST. She is a member of IMS as well. Her research is the analysis, design, and application of digital signatures, secret sharing schemes, security protocols, etc.