ON PROXY (MULTI-) SIGNATURE SCHEMES[1]

Hung-Min Sun

Department of Computer Science and Information Engineering
National Cheng Kung University, Tainan, Taiwan 70101
Email: hmsun@mail.ncku.edu.tw

ABSTRACT

A proxy signature scheme allows a designed person, called a proxy signer, to sign messages on behalf of an original signer. Generalizing the concept of the proxy signature scheme, Yi *et al.* proposed the proxy multi-signature scheme which allows a proxy signer to generate a proxy signature on behalf of two or more original signers. In this paper, we first analyze and improve the security of two proxy signature schemes, proposed by Sun and Hsieh at IS'99. Our analysis indicates that these two schemes suffer from the public key substitution attack and a kind of direct forgery. Then we show that the same attacks on Sun-Hsieh proxy signature schemes can be generalized to work on Yi *et al.*'s proxy multi-signature schemes. Two proxy multi-signature schemes are consequently proposed to defeat these attacks. Finally, we point out that the Sun-Lee-Hwang threshold proxy signature scheme is also vulnerable to the same attacks above. An improved version is therefore proposed.

1. INTRODUCTION

The concept of the proxy signature scheme was first introduced by Mambo *et al.* [1] in 1996. A proxy signature scheme allows a designed person, called a proxy signer, to sign on behalf of an original signer. In [2], Mambo *et al.* further proposed a more secure version in which the proxy signer cannot repudiate the creation of a valid proxy signature against anyone later. This property is usually referred to as "nonrepudiation". So far, a number of proxy signature schemes with nonrepudiation property have been constructed [1-5]. Part of them are named proxy-protected proxy signature schemes [2-3] and the others are named nonrepudiable proxy signature schemes [4-5]. Among these nonrepudiable proxy-protected proxy signature schemes, Zhang's scheme [4] has been shown to be insecure due to Lee et al. [6]. Recently, Sun and Hsieh [5] showed that the Mambo-Usuda-Okamoto scheme is unfair to the original signer because the proxy signer can transfer the delegation to others, and that the Kim-Park-Won scheme is vulnerable to the public key substitution attack in which an attacker can forge a valid proxy signature by updating his own public key. To repair both the Mambo-Usuda-Okamoto scheme and the Kim-Park-Won scheme, they also presented two modified versions in [5].

Generalizing the concept of the proxy signature, Yi *et al.* [7] proposed a new type of proxy signature scheme, named proxy multi-signature scheme, in which a proxy signer can generate a proxy signature on behalf of two or more original signers. They proposed two proxy multi-signature schemes based on the Mambo-Usuda-Okamoto proxy signature scheme and the Kim-Park-Won proxy signature scheme respectively.

On the other hand, a (t, n) threshold proxy signature scheme [3,4,8] is a variant of the proxy signature scheme in which the proxy signature key is shared by a group of n proxy signers in such a way that any t or more proxy signers can cooperatively employ the proxy signature key to sign messages on behalf of an original signer, but t-1 or fewer proxy signers cannot. In [8], Sun, Lee, and Hwang showed that the threshold proxy signature scheme proposed by Zhang [4] suffers from some weaknesses and the threshold proxy signature scheme proposed by Kim *et al.* [3] suffers from a disadvantage. In addition, they also proposed a new threshold proxy signature scheme, the Sun-Lee-Hwang scheme in short, to prevent these weaknesses.

In this paper, we first analyze and improve the security of the two proxy signature schemes, proposed by Sun and Hsieh. Our analysis indicates that these two schemes suffer from the public key substitution attack and a kind of direct forgery. Then we show that the same attacks on Sun-Hsieh proxy signature schemes can be generalized to work on Yi *et al.*'s proxy multi-signature schemes. Two proxy multi-signature schemes are consequently proposed to defeat these attacks. Finally, we point out that the Sun-Lee-Hwang threshold proxy signature scheme is also vulnerable to the same attacks above. An improved version is therefore proposed.

[1] This research was supported in part by the National Science Council, Taiwan, under contract NSC-89-2213-E-006-118.

The remainder of this paper is organized as follows. In section 2, we analyze and improve the Sun-Hsieh proxy signature scheme based on the Mambo-Usuda-Okamoto scheme. In section 3, we analyze and improve the Sun-Hsieh proxy signature scheme based on the Kim-Park-Won scheme. In section 4 and 5, we analyze the Mambo-like proxy multi-signature scheme and the Kim-like proxy multi-signature scheme, proposed by Yi *et al.* In section 6 and 7, we further propose a proxy-unprotected proxy multi-signature scheme and a proxy-protected proxy multi-signature scheme against forgery respectively. In section 8, we analyze and improve the Sun-Lee-Hwang threshold proxy signature. Section 9 gives a general approach to defeat the public key substitution attack. Finally, we conclude this paper in section 10.

2. ON SUN-HSIEH PROXY SIGNATURE SCHEME BASED ON MAMBO-USUDA-OKAMOTO SCHEME

2.1 Description of the Scheme [5]

Let p be a large prime and g be a generator for Z_p^*. The original signer has a private key $s_o \in Z_{p-1}/\{0\}$ and the corresponding public key $v_o = g^{s_o}$ (mod p); the proxy signer has a private key $s_p \in Z_{p-1}/\{0\}$ and the corresponding public key $v_p = g^{s_p}$ (mod p). Both v_o and v_p are certified by a certification authority (CA). Let $h(\ )$ be a public collision resistant hash function. For simplicity, we describe only the signature verification process for a proxy signature.

A proxy signature, generated by the proxy signer, on a message m is a 4-tuple ($m, Sign_{\sigma_p}(m), K$, M_w), where $Sign_{\sigma_p}(m)$ is the signature on m using an ordinary signature scheme with a proxy signature key σ_p. The validity of the proxy signature can be checked by the verification equation in the ordinary signature scheme with the corresponding public key

$$v = v_o \cdot {v_p}^{v_p} K^{h(M_w,K)} \pmod p .$$

2.2 Cryptanalysis of the Scheme

In this section, we show that the original signer can make the public key substitution attack feasible. First, the original signer selects a random number $k \in Z_{p-1}/\{0\}$, and computes $K = g^k$ (mod p). Then he selects a random number $a \in Z_{p-1}/\{0\}$ and updates his pulic key by v_o = $({v_p}^{v_p} K^{h(M_w,K)})^{-1} \cdot g^a$ (mod p). Thus $\sigma_p = a$ is a valid proxy signature key. This is because

$v = v_o \cdot {v_p}^{v_p} K^{h(M_w,K)}$ (mod p)

$= ({v_p}^{v_p} K^{h(M_w,K)})^{-1} \cdot g^a \cdot {v_p}^{v_p} K^{h(M_w,K)}$ (mod p)

$= g^a$ (mod p) $= g^{\sigma_p}$ (mod p).

2.3 Our Improvement

Here we present an improved proxy signature scheme to defeat the above forgery.

Step I (Proxy generation): The original signer first chooses a random number $k \in Z_{p-1}/\{0\}$, and then computes $K = g^k$ (mod p) and $\sigma = s_o \cdot v_o + k \cdot h(M_w, K)$ (mod p-1), where M_w is a warrant which contains the original signer's ID, the proxy signer's ID, the delegation period, the issue time for the delegation.

Step II (Proxy delivery): The original signer sends (σ, K, M_w) to the proxy signer over a public channel.

Step III (Verification and alteration of the proxy): The proxy signer confirms the validity of (σ, K, M_w) by checking if the following congruence holds:

$$g^{\sigma} = {v_o}^{v_o} K^{h(M_w,K)} \pmod p.$$

If it holds, then the proxy signer computes an alternative proxy signature key

$$\sigma_p = \sigma + s_p \cdot h(M_w, K, v_p) \mod (p\text{-}1).$$

Step IV (Signing by the proxy signer): The proxy signer signs a message m by using an ordinary signature scheme with secret key σ_p. Assume that the resulting signature is $Sign_{\sigma_p}(m)$. The proxy signature on m is ($m, Sign_{\sigma_p}(m), K$, M_w).

Step V (Verification of the proxy signature): The verifier computes the corresponding public key in the ordinary signature scheme:

$$v = {v_o}^{v_o} \cdot {v_p}^{h(M_w,K,v_p)} K^{h(M_w,K)} \pmod p .$$

Then, he verifies the validity of $Sign_{\sigma_p}(m)$ by checking the validity of the verification equation in the ordinary signature scheme with the new generated public key v.

3. ON SUN-HSIEH PROXY SIGNATURE SCHEME BASED ON KIM-PARK-WON SCHEME

3.1 Description of the Scheme [5]

The system parameters are the same as those in Section 2.1. For simplicity, we describe only the signature verification process for a proxy signature. Similar to Section 2.1, a proxy signature on a message *m* is a 4-tuple ($m, Sign_{\sigma_p}(m), K,\ M_w$). The validity of the proxy signature can be checked by the verification equation in the ordinary signature scheme with the corresponding public key $v = v_o^{h(M_w,K,v_p)} \cdot v_p \cdot K \text{ (mod } p)$.

3.2 Cryptanalysis of the Scheme

In the following, we show that (i) the original signer can make the public key substitution attack feasible; (ii) the original signer can create a valid proxy signature key σ_p with respect to an arbitrary user (who has/hasn't been told as the proxy signer).

(i) First, the original signer selects a random number $k \in Z_{p-1}/\{0\}$, and computes $K = g^k$ (mod *p*) and $e = h(M_w, K, v_p)$. Then he selects a random number $a \in Z_{p-1}/\{0\}$ and updates his pulic key by $v_o = v_p^{-e^{-1}} \cdot g^a$ (mod *p*). Finally, he computes $\sigma_p = ae + k$ (mod *p*-1). Thus σ_p is a valid proxy signature key. This is because $v = v_o^{h(M_w,K,v_p)} \cdot v_p \cdot K \text{ (mod } p) = (v_p^{-e^{-1}} g^a)^e \cdot v_p \cdot g^k$ (mod *p*) = g^{σ_p} (mod *p*).

(ii) First, the original signer randomly selects a number $k \in Z_{p-1}/\{0\}$, and computes $K = g^k \cdot v_p^{-1}$ (mod *p*). Then he computes $\sigma_p = h(M_w, K, v_p) \cdot s_o + k$ (mod *p*-1). Thus σ_p is a valid proxy signature key because $v = v_o^{h(M_w,K,v_p)} \cdot v_p \cdot K \text{ (mod } p)$ (mod *p*)= $g^{h(M_w,K,v_p)\cdot s_o + k}$ (mod *p*) = g^{σ_p} (mod *p*).

3.3 Our Improvement

In this section, we present an improved proxy signature scheme to defeat the above forgery.

Step 1 (Proxy generation): The original signer first chooses a random number $k \in Z_{p-1}/\{0\}$, and then computes $K = g^k$ (mod *p*), $e = h(M_w, K, v_p)$, and $\sigma = es_o + k$ (mod *p*-1).

Step 2 (Proxy delivery): The original signer sends (M_w, σ, K) to the proxy signer over a public channel.

Step 3 (Verification and alteration of the proxy): The proxy signer confirms the validity of (M_w, σ, K) by checking if the following congruence holds:

$$g^{\sigma} = v_o^{h(M_w,K,v_p)} \cdot K \text{ (mod } p).$$

If it holds, then the proxy signer computes an alternative proxy signature key

$\sigma_p = \sigma + s_p\ h(M_w, v_o, K)$ mod (*p*-1).

Step 4 (Signing by the proxy signer): The proxy signer signs a message *m* by using an ordinary signature scheme with secret key σ_p. Assume that the resulting signature is $Sign_{\sigma_p}(m)$. The proxy signature on *m* is ($m, Sign_{\sigma_p}(m), K,\ M_w$).

Step 5 (Verification of the proxy signature): The verifier computes the corresponding public key in the ordinary signature scheme:

$v = v_o^{h(M_w,K,v_p)} \cdot v_p^{h(M_w,v_o,K)} \cdot K \text{ (mod } p)$.

Then, he verifies the validity of $Sign_{\sigma_p}(m)$ by checking the validity of the verification equation in the ordinary signature scheme with the new generated public key *v*.

4. ON MAMBO-LIKE PROXY MULTI-SIGNATURE SCHEME

4.1 Description of the Scheme [7]

Let $A_1, A_2, \ldots, A_n$ be *n* original signers with private key $s_i \in Z_{p-1}/\{0\}$ and public key $v_i = g^{s_i}$ (mod *p*) respectively. These v_i are certified by a CA.

Step A (Subproxy key generation): For each 1≤ *i* ≤*n*, the original signer A_i chooses a random number $k_i \in Z_{p-1}/\{0\}$, and then computes $K_i = g^{k_i}$ (mod *p*) and $\sigma_i = s_i + k_i K_i$ (mod *p*-1).

Step B (Subproxy key delivery): For each 1≤ i ≤n, the original signer A_i sends (σ_i, K_i) to the proxy signer in a secure manner.

Step C (Subproxy key verification): For each 1≤ i ≤n, the proxy signer confirms the validity of (σ_i, K_i) by checking if the following congruence holds: $g^{\sigma_i} = v_i K_i^{K_i}$ (mod p).
If (σ_i, K_i) passes this equation, he accepts it as a valid subproxy key; otherwise, he rejects it and requests A_i for a valid one, or he terminates this protocol.

Step D (Proxy signature key generation): If the proxy signer confirms the validity of all (σ_i, K_i) for 1≤ i ≤n, then he computes $\sigma_p = \sum_{i=1}^{n} \sigma_i$ mod (p-1) as a valid proxy signature key.

Step E (Signing by the proxy signer): The proxy signer executes the signing operation of an ordinary signature scheme using σ_p as the signing key. Assume that the resulting signature is $Sign_{\sigma_p}(m)$. The proxy multi-signature on m for A_1, A_2, …, A_n is ($m, Sign_{\sigma_p}(m), K_1, ..., K_n$).

Step F (Verification of the proxy multi-signature): The verifier computes the corresponding proxy public key in the ordinary signature scheme:

$$v = v_1 \cdots v_n K_1^{K_1} \cdots K_n^{K_n} \text{ (mod } p).$$

Then, he verifies the validity of $Sign_{\sigma_p}(m)$ by checking the validity of the verification equation in the ordinary signature scheme with the new generated proxy public key v.

4.2 Cryptanalysis of the Scheme

In the following, we show that the Mambo-like proxy multi-signature scheme is also insecure against the public key substitution attack that an attacker can forge a valid proxy multi-signature by updating his own public key. Without loss of generality, we assume that A_1 wants to forge a proxy multi-signature on m for A_1, A_2, …, A_t. He first selects t+1 random numbers, σ_p, $K_1, ...,$ and $K_t \in Z_{p-1}/\{0\}$, and then computes $v_1{*} = (v_2 \cdots v_t \cdot K_1^{K_1} \cdots K_t^{K_t})^{-1} g^{\sigma_p}$ (mod p). Then he makes a request to CA for updating his public key v_1 with $v_1{*}$ (Note that he may claim that he has lost his private key). Thus σ_p is a valid proxy signature key and v= g^{σ_p} (mod p) is the corresponding proxy public key.
This is because $v = v_1 {*} \cdots v_t K_1^{K_1} \cdots K_t^{K_t}$ (mod p) = $(v_2 \cdots v_t \cdot K_1^{K_1} \cdots K_t^{K_t})^{-1} g^{\sigma *} (v_2 \cdots v_t \cdot K_1^{K_1} \cdots K_t^{K_t})^{-1} = g^{\sigma_p}$ (mod p). Therefore, A_1 can use σ_p to generate a forged proxy multi-signature on an arbitrary message m for A_1, A_2, …, A_t.

5. ON KIM-LIKE PROXY MULTI-SIGNATURE SCHEME

5.1 Description of the Scheme [7]

The system parameters are the same as those in the Mambo-like proxy multi-signature scheme.

Step i (Subproxy key generation): is the same as **Step A** except that $\sigma_i = e_i s_i + k_i$ (mod p-1), where $e_i = h(M_w, K_i)$.

Step ii (Subproxy key delivery): is the same as **Step B** except that (σ_i, K_i) is replaced with (M_w, σ_i, K_i).

Step iii (Subproxy key verification): is the same as **Step C** except that $g^{\sigma_i} = v_i^{e_i} K_i$ (mod p), where $e_i = h(M_w, K_i)$.

Step iv (Proxy signature key generation): is the same as **Step D** except that (σ_i, K_i) is replaced with (M_w, σ_i, K_i).

Step v (Signing by the proxy signer): is the same as **Step E** except that the proxy multi-signature on m for A_1, A_2, …, A_n is ($m, Sign_{\sigma_p}(m), K_1, ..., K_n, M_w$).

Step vi (Verification of the proxy multi-signature): is the same as **Step F** except that $v = v_1^{e_1} \cdots v_n^{e_n} K_1 \cdots K_n$ (mod p), where $e_i = h(M_w, K_i)$ for 1≤ i ≤n.

5.2 Cryptanalysis of the Scheme

In the following, we show that the Kim-like proxy multi-signature scheme is insecure against a direct forgery. Without loss of generality, we assume that A_1 wants to forge a proxy multi-signature on m for A_1, A_2, …, A_t. He first selects t-1 random numbers, $K_2, ..., K_t \in Z_{p-1}/\{0\}$, and then computes $e_i = h(M_w, K_i)$ for 2≤ i ≤ t. Then he

selects a random number $k_1 \in Z_{p-1}/\{0\}$, computes $K_1 = (v_2^{e_2} \cdots v_t^{e_t} K_2 \cdots K_t)^{-1} g^{k_1}$ (mod *p*) and $e_1 = h(M_w, K_1)$. Let s_1 be the private key of A_1. Thus $\sigma_p = s_1 \cdot e_1 + k_1$ (mod *p*-1) is a valid proxy signature key and *v*= g^{σ_p} (mod *p*) is the corresponding proxy public key. This is because
$v = v_1^{e_1} \cdots v_t^{e_t} K_1 \cdots K_t$ (mod *p*) =
$v_1^{e_1} (v_2^{e_2} \cdots v_t^{e_t} K_2 \cdots K_t) K_1$ (mod *p*) = $(g^{s_1})^{e_1} g^{k_1}$ (mod *p*) = g^{σ_p} (mod *p*). Therefore, A_1 can use σ_p to generate a forged proxy multi-signature on an arbitrary message *m* for A_1, A_2, …, A_t.

6. PROXY-UNPROTECTED PROXY MULTI-SIGNATURE SCHEME AGAINST FORGERY

Both the Mambo-like and the Kim-like proxy multi-signature schemes, proposed by Yi *et al.*, are proxy-unprotected because the identity of the proxy signer cannot be proved. In this section, we present a proxy-unprotected proxy multi-signature scheme which is secure against forgery. The system parameters are the same as those in the Mambo-like multi-signature scheme and we assume that the proxy signer has a private key $s_p \in Z_{p-1}/\{0\}$ and the corresponding public key $v_p = g^{s_p}$ (mod *p*).

Step a (Subproxy key generation): is the same as **Step A** except that $\sigma_i = s_i \cdot v_i + k_i K_i$ (mod *p*-1).

Step b (Subproxy key delivery): is the same as **Step B**.

Step c (Subproxy key verification): is the same as **Step C** except that $g^{\sigma_i} = v_i^{v_i} K_i^{K_i}$ (mod *p*).

Step d (Proxy signature key generation): is the same as **Step D**.

Step e (Signing by the proxy signer): is the same as **Step E**.

Step f (Verification of the proxy multi-signature): is the same as **Step F** except that $v = v_1^{v_1} \cdots v_n^{v_n} K_1^{K_1} \cdots K_n^{K_n}$ (mod *p*).

Security Analysis:

Without loss of generality, we assume *n*=2. Therefore, $v = v_1^{v_1} v_2^{v_2} K_1^{K_1} K_2^{K_2}$ (mod *p*).
In the following, we show that the public key substitution attack cannot work here. Without loss of generality, we assume that A_1 wants to forge a proxy multi-signature on *m* for A_1 and A_2. Let $v_1 = v_2^{a}.g^{b}$ (mod *p*), $K_1 = v_2^{c}.g^{d}$ (mod *p*), and $K_2 = v_2^{e}.g^{f}$ (mod *p*). Then $g^{\sigma_p} = v_2^{av_1+v_2+cK_1+eK_2} g^{bv_1+dK_1+fK_2}$ (mod *p*) must hold. If A_1 can make the equation $av_1 + v_2 + cK_1 + eK_2 = 0$ (mod *p*-1) hold, he can obtain $\sigma_p = bv_1 + dK_1 + fK_2 = 0$ (mod *p*-1). However, it is infeasible to find *a*, *c*, and *e* such that $av_1 + v_2 + cK_1 + eK_2 = 0$ (mod *p*-1) holds. This is because v_1 depends on *a*, K_1 depends on *c*, and K_2 depends on *e*, and hence any change of *a*, *c*, and *e* will lead to the change of v_1, K_1, and K_2 respectively. If he fixes v_1, K_1, and K_2 such that they are independent of *a*, *c*, and *e* respectively, then he must solve the difficult discrete logarithm problem to find *b*, *d*, and *f*.

7. PROXY-PROTECTED PROXY MULTI-SIGNATURE SCHEME AGAINST FORGERY

In Yi *et al.*'s paper, they mentioned that their idea can also be used to construct proxy-protected proxy multi-signature schemes in which the proxy signer cannot later repudiate the proxy multi-signature which he has ever signed. Similarly, these constructed proxy-protected proxy multi-signature schemes also suffer from the same security problems as described above. In this section, we present a proxy-protected proxy multi-signature scheme which is secure against forgery. The system parameters are the same as those in the Mambo-like proxy multi-signature scheme. Moreover, let *h*( ) be a public collision resistant hash function. Furthermore, the proxy signer has a private key $s_p \in_R Z_{p-1}/\{0\}$ and a public key $v_p = g^{s_i}$ (mod *p*).

Step (1) (Subproxy key generation): is the same as **Step A** except that $\sigma_i = s_i \cdot v_i + k_i h(M_w, K_i)$ (mod *p*-1), where M_w is a warrant which contains the original signer ID, the proxy signers' ID, the issue time for the delegation, the valid delegation period, etc..

Step (2) (Subproxy key delivery): For each 1≤ *i* ≤*n*, the original signer A_i sends (M_w, σ_i, K_i) to the proxy signer over a public channel.

Step (3) (Subproxy key verification): is the same as **Step C** except that $g^{\sigma_i} = v_i^{v_i} K_i^{h(M_w, K_i)}$ (mod *p*).

Step (4) (Proxy signature key generation): If the proxy signer confirms the validity of all (M_w, σ_i, K_i) for 1≤ *i* ≤*n*, then he computes

$\sigma_p = s_p \cdot v_p + \sum_{i=1}^{n} \sigma_i$ mod (p-1) as a valid proxy signature key.

Step (5) (Signing by the proxy signer): is the same as **Step E** except that the proxy multi-signature on m for A_1, A_2, …, A_n is $(m, Sign_{\sigma_p}(m), K_1, ..., K_n, M_w)$.

Step (6) (Verification of the proxy multi-signature): is the same as **Step F** except that

$$v = v_p{}^{v_p} \cdot v_1{}^{v_1} \cdots v_n{}^{v_n} \cdot K_1{}^{h(M_w, K_1)} \cdots K_n{}^{(M_w, K_n)} \pmod p .$$

Security Analysis:

The difference between the proxy-protected proxy multi-signature scheme and the proxy-unprotected proxy multi-signature scheme proposed in Section 6 is that the proxy signer's public key is included in the proxy public key v, and $K_i{}^{K_i}$ is replaced by $K_i{}^{h(M_w, K_i)}$. Similar to Section 6, the proposed scheme is also secure against the public key substitution attack. Here we also note why $K_i{}^{K_i}$ is replaced by $K_i{}^{h(M_w, K_i)}$. If $K_i{}^{K_i}$ is used, the resulting scheme is unfair to the original signers because the proxy signer can transfer the delegation to others. The use of $K_i{}^{h(M_w, K_i)}$ limits the transference because M_w indicates who the proxy signer is.

8. ON SUN-LEE-HWANG THRESHOLD PROXY SIGNATURE SCHEME

In this section, we show that the Sun-Lee-Hwang threshold proxy signature scheme [8] is also vulnerable to the public key substitution attack and a direct forgery. For simplicity, we describe only the signature verification process of the Sun-Lee-Hwang scheme.

Let p be a large prime, q be a prime factor q of p-1, and g be an element of order q in Z_p^*. Let p_1, p_2, …, p_n be the n proxy signers. Assume that the original signer has a private key x_o and a public key $y_o = g^{x_o}$ (mod p); and each proxy signer p_i has a private key x_i and a public key $y_i = g^{x_i}$ (mod p). All these public keys are certified by a CA. The *PGID* (Proxy Group ID) which records the proxy status is defined to be {*EM*, *Time*, *Group*}, where *EM* denotes the event mark of the proxy share generation including the parameters t and n, *Time* denotes the valid delegation period, and *Group* denotes the information describing the identities of the original signer and the proxy signers of the group. A threshold proxy signature on m is (m, r, *PGID*, Y, T), which is cooperatively generated by t out of n proxy signers. Any verifier can verify the validity of the threshold proxy signature (m, r, *PGID*, Y, T) by checking if the following equation holds:

$$g^T = [y_o{}^{h(r,PGID)} r \prod_{i=1}^{n} y_i\ ]^{h(m)}\ Y^Y \pmod p.$$

8.1 Cryptanalysis

Without loss of generality, we assume that p_1 wants to forge a threshold proxy signature on an arbitrary message m for an arbitrary proxy group { p_1, p_2, …, p_n }. He first selects three random numbers, r, a and $b \in Z_q$, and a forged *PGID*. Then he computes $Y = g^b$ (mod p), $y_1 = (y_o{}^{h(r,PGID)} r \prod_{i=2}^{n} y_i)^{-1} g^a$ (mod p), and $T = ah(m) + bY$ (mod q). Finally, he makes a request to CA for updating his public key with y_1. Thus (m, r, *PGID*, Y, T) is a valid threshold proxy signature because the verification equation, $g^T = [y_o{}^{h(r,PGID)} r \prod_{i=1}^{n} y_i\ ]^{h(m)}\ Y^Y$ (mod p), holds.

In addition, the original signer can also forge a valid threshold proxy signature with respect to an arbitrary proxy group by updating his own public key, while the proxy group can not repudiate. We describe this forgery as follows:

We assume that the original signer wants to forge a threshold proxy signature on an arbitrary message m for an arbitrary proxy group { p_1, p_2, …, p_n }. He first selects three random numbers, r, a and $b \in Z_q$, and a forged *PGID*. Then he computes $Y = g^b$ (mod p), $y_o = (r \prod_{i=1}^{n} y_i)^{-h(r,PGID)^{-1}} g^a$ (mod p), and $T = a \cdot h(r,PGID) \cdot h(m) + b \cdot Y$ (mod q). Finally, he makes a request to CA for updating his public key with y_1. Thus (m, r, *PGID*, Y, T) is a valid threshold proxy signature because the verification equation, $g^T = [y_o{}^{h(r,PGID)} r \prod_{i=1}^{n} y_i\ ]^{h(m)}\ Y^Y$ (mod p), holds.

In the following, we show that a direct forgery by the original signer can work. The original signer first selects two random numbers a and $b \in Z_q$, and a forged *PGID*. Then he computes $Y = g^a$ (mod p), $r = (\prod_{i=1}^{n} y_i)^{-1} g^b$ (mod p), and $T = [x_o \cdot h(r, PGID) + b] \cdot h(m) + a \cdot Y$ (mod q). Thus (m, r, *PGID*, Y, T) is a valid threshold proxy signature because the verification equation, g^{T} = $[y_o^{h(r,PGID)} r \prod_{i=1}^{n} y_i\ ]^{h(m)}\ Y^{Y}$ (mod p), holds.

8.2 Our Improvement

In this section, we present an improved threshold proxy signature scheme in order to defeat the above attacks.

Proxy share generation:

Step (I) The original signer randomly selects $\tilde{k} \in Z_q$, computes $\tilde{r} = g^{\tilde{k}}$ (mod p), and broadcasts $\tilde{r}$.

Step (II) (a) Each proxy signer p_i randomly selects $\alpha_i \in Z_q$ and computes $r_i = g^{\alpha_i}\tilde{r}$ (mod p).
(b) Each proxy signer p_i checks whether $r_i \in Z_p^{*}$. If this is not true, he goes back to step (a). Otherwise, he broadcasts r_i.

Step (III) The original signer computes $r = \Pi_{i=1}^{n} r_i$, $\tilde{s} = n^{-1}x_o h(r, PGID, y_o) + \tilde{k}$ (mod q), and broadcasts $\tilde{s}$.

Step (IV) Each proxy signer p_i computes $r = \Pi_{i=1}^{n} r_i$ and checks if the following equation holds: $g^{\tilde{s}} = y_o^{n^{-1}h(r,PGID,y_o)}\tilde{r}$ (mod p).
If it doesn't hold, p_i broadcasts an error and stops.
Each proxy signer p_i computes $s_i = \tilde{s} + \alpha_i + x_i \cdot h(r, y_i)$ (mod q).

Step (V) Each proxy signer p_i randomly selects a polynomial f_i of degree t-1 in Z_q such that $f_i(0) = s_i$. That is $f_i(x) = s_i + a_{i,1}x + \ldots + a_{i,t-1}x^{t-1}$ (mod q). Then p_i sends $f_i(j)$ mod q to p_j (for 1≤ j ≤n and j≠i) in a secure method. In addition, p_i also broadcasts $g^{a_{i,1}}, \ldots, g^{a_{i,t-1}}$ [Note that g^{s_i} doesn't need to be broadcasted here because g^{s_i} can be computed by: $g^{s_i} = g^{\tilde{s}} r_i y_i^{h(r,y_i)} \tilde{r}^{-1}$. For each distributed $f_j(i)$ (for 1≤ j ≤n and j≠i), p_i can verify the validity of $f_j(i)$ by checking if the following equation holds:

$g^{f_j(i)} = g^{\tilde{s}} r_j y_j^{h(r,y_j)} \tilde{r}^{-1} (g^{a_{j,1}})^{i} (g^{a_{j,2}})^{i^2} \cdots (g^{a_{j,t-1}})^{i^{t-1}}$ (mod p).

If all $f_j(i)$ are verified to be legal, each p_i computes $x_i' = \sum_{j=1}^{n} f_j(i)$ (mod q) as a valid proxy share. Let $f(x) = \sum_{j=1}^{n} f_j(x) = \sum_{j=1}^{n} s_j + (\sum_{j=1}^{n} a_{j,1})x + \ldots + (\sum_{j=1}^{n} a_{j,t-1})x^{t-1}$ (mod q).
Hence, $x_i' = f(i)$.

Note that in **Step (V)**, if these n proxy signers collude, they may change the threshold value t into t'. However, this collusion is not meaningful because t dishonest proxy signers can sign any message that they want to sign.

Generation of the proxy signature without revealing shares :

Without loss of generality, we assume that $p_1, \ldots, p_t$ are the t proxy signers who want to cooperate to sign a message m on behalf of the original signer. Each proxy signer p_i (1≤ i ≤t) randomly selects a polynomial f_i' of degree t-1 in Z_q. That is $f_i'(x) = a_{i,0}' + a_{i,1}'x + \ldots + a_{i,t-1}'x^{t-1}$ (mod q). Thus $f_i'(0) = a_{i,0}'$. Then p_i sends $f_i'(j)$ to p_j (for 1≤ j ≤t and j≠i) in a secure method. In addition, p_i also broadcasts $g^{a_{i,0}'}, g^{a_{i,1}'}, \ldots, g^{a_{i,t-1}'}$. For each distributed $f_j'(i)$ (for 1≤ j ≤t and j≠i), p_i can verify the validity of $f_j'(i)$ by checking if the following equation holds:

$g^{f_j'(i)} = g^{a_{j,0}'} (g^{a_{j,1}'})^{i} (g^{a_{j,2}'})^{i^2} \cdots (g^{a_{j,t-1}'})^{i^{t-1}}$ (mod p).

If all $f_j'(i)$ are verified to be legal, each p_i computes $x_i'' = \sum_{j=1}^{t} f_j'(i)$ (mod q) and $Y = \prod_{j=1}^{t} g^{a_{j,0}'}$ (mod p).

Let $f'(x) = \sum_{j=1}^{t} f_j'(x) = \sum_{j=1}^{t} a_{j,0}' + (\sum_{j=1}^{t} a_{j,1}')x + \ldots + (\sum_{j=1}^{t} a_{j,t-1}')x^{t-1}$ (mod q). Thus, $x_i'' = f'(i)$. Then each p_i computes $T_i = x_i' h(m) + x_i'' Y$ (mod q). Let $f''(x) = f(x)h(m) + f'(x)Y$. Hence $T_i = f''(i)$ (mod q). Each p_i broadcasts T_i. For each distributed T_j (for 1≤ j ≤t and j≠i), p_i can verify the validity of T_j by checking if the following equation holds:

$$g^{T_i} = [\, g^{n\tilde{s}}\tilde{r}^{-n} \prod_{j=1}^{n} r_j \prod_{j=1}^{n} y_j^{h(r,y_j)} \ (\prod_{j=1}^{n} g^{a_{j,1}})^{i} \ (\prod_{j=1}^{n} g^{a_{j,2}})^{i^2} \cdots$$

$$(\in_{j-1}^{n} g^{a_{j,t-1}})^{i^{t-1}} \ ]^{h(m)} \cdot [\, (\in_{j-1}^{t} g^{a_{j,0}'}) \ (\in_{j-1}^{t} g^{a_{j,1}'})^{i} \ (\in_{j-1}^{t} g^{a_{j,2}'})^{i^2} =$$

$$(\in_{j-1}^{t} g^{a_{j,t-1}'})^{i^{t-1}} \ ]^{Y} \ (\text{mod } p).$$

Each p_i computes $T = f''(0) = f(0)h(m) + f'(0)Y$ from T_j (for $1+ j +t$) by applying Lagrange interpolating polynomial. The proxy signature of m is (m, r, $PGID$, Y, T).

Verification of the proxy signature:

Because $g^{f(0)}$ (mod p) = $g^{\leq_{i=1}^{n} s_i}$ (mod p) =

$g^{n\tilde{s}\Sigma\leq_{i=1}^{n}(\sigma_i \Sigma x_i h(r,y_i))}$ (mod p) =

$g^{h(r,PGID,y_o)x_o\Sigma\leq_{i=1}^{n}(\sigma_i \Sigma\tilde{k}\Sigma x_i h(r,y_i))}$ (mod p) =

$y_o^{h(r,PGID,y_o)} \in_{i-1}^{n} r_i \in_{i-1}^{n} y_i^{h(r,y_i)}$ (mod p) =

$y_o^{h(r,PGID,y_o)} r \in_{i-1}^{n} y_i^{h(r,y_i)}$ (mod p), we can use $g^{f(0)}$ as the new public key.

Thus verifier can verify the validity of the proxy signature (m, r, $PGID$, Y, T) by checking if the following equation holds: $g^{T} = [y_o^{h(r,PGID,y_o)} r \in_{i-1}^{n} y_i^{h(r,y_i)}]^{h(m)} Y^{Y}$ (mod p),

where y_o is the original signer's public key and y_i's are the proxy signers' public keys.

9. A GENERAL APPROACH TO DEFEAT THE PUBLIC KEY SUBSTITUTION ATTACK

From Section 2 and 4, we know that both the Sun-Hsieh proxy signature scheme based on the Mambo-Usuda-Okamoto scheme and the Mambo-like proxy multi-signature scheme are only subject to the public key substitution attack. In this section, we propose a general approach to defeat the public key substitution attack without modifying these two schemes.

It is clear that under the public key substitution attack, an attacker must update his public key, but the corresponding private key is unknown to him due to the difficulty of the discrete logarithm problem. Therefore we need a strict CA that can prove the user's knowledge of the corresponding private key when a user registers or updates his public key. This work may be completed by either encryption or signature. For example, the CA selects a challenge number r, encrypts it with the user's public key and sends the ciphertext to the user. The user must response r to CA in order to prove that he knows the corresponding private key. On the other hand, the CA may request the user to sign a challenge number r, and then check the validity of the signature on r with the user's public key.

10. CONCLUSIONS

In this paper, we have shown that some proxy signature, proxy multi-signature, and threshold proxy signature schemes are vulnerable to the public key substitution attack and/or the direct forgery. The improved versions of these schemes are proposed to defeat these attacks. Due to the limit of space, we omit the security analysis of these improved schemes. The detailed analysis for these improved versions are given in the full version of this paper.

REFERENCES

[1] M. Mambo, K. Usuda, E. Okamoto, "Proxy signatures for delegating signing operation," Proc. 3rd ACM Conference on Computer and Communications Security, 1996, pp. 48-57.

[2] M. Mambo, K. Usuda, E. Okamoto, "Proxy signatures: Delegation of the power to sign messages," *IEICE Trans. Fundamentals*, 1996, E79-A, (9), pp. 1338-1354.

[3] S. Kim, S. Park, and D. Won, "Proxy signatures, revisited," ICICS'97, Lecture Notes in Computer Science Vol. 1334, (Springer-Verlag, 1997), pp. 223-232.

[4] K. Zhang, "Threshold proxy signature schemes," *1997 Information Security Workshop*, Japan, Sep., 1997, pp. 191-197.

[5] H.-M. Sun and B.-T. Hsieh, "Remarks on two nonrepudiable proxy signature schemes," *Proceeding of Ninth National Conference on Information Security*, pp. 241-246. 1999.

[6] N.-Y. Lee, T. Hwang, and C.-H. Wang, "On Zhang's nonrepudiable proxy signature schemes," ACISP'98, Lecture Notes in Computer Science, Vol. 1438, Springer-Verlag, 1998, pp. 415-422.

[7] L. Yi, G. Bai, and G. Xiao, "Proxy multi-signature scheme: A new type of proxy signature scheme," *Electronics Letters*, 2000, Vol. 36, No. 6, pp. 527-528.

[8] H.-M. Sun, N.-Y. Lee, and T. Hwang, "Threshold proxy signatures," *IEE Proceedings - Computers and Digital Techniques*, Vol. 146, No. 5, pp. 259-263, 1999.