Improvement of Threshold Signature Using Self-certified Public Keys

Zuhua Shao

Department of Computer and Electronic Engineering, Zhejiang University of Science and Technology, No. 85, XueYuan Road, Hangzhou, Zhejiang, P. R. of China, 310012 (Email: zhshao_98@yahoo.com)

(Received Feb. 18, 2005; revised and accepted March 18, 2005)

Abstract

A (t, n) threshold signature scheme allows any t or more signers to cooperatively sign messages on behalf of a group, but $t-1$ or fewer signers cannot. Wu and Hsu recently proposed a new (t, n) threshold signature scheme using self-certified public keys. In their scheme, the authentication of the self-certified individual/group public keys can be confirmed simultaneously in the procedure of verifying the individual/group signatures. Compared with threshold signature schemes based on the certified-based public key systems, their scheme is more efficient. However, the author of this paper points out that there are some problems in the Wu-Hsu scheme. The registration stage cannot work since there is a deadlock in the computation of the self-certified individual public keys. Moreover, some t or more malicious signers can conspire together against the group, and the system authority SA can also conspire with a malicious user to forge the group public key without being detected. Finally, we propose an improvement to counter the two attacks. Its signature computation and verification are more efficient than that of the Wu-Hsu scheme. The system authority SA can identify the actual signers, while they are anonymous to outsiders.

Keywords: Coalition attack, cryptography, self-certified, threshold signature

1 Introduction

Digital signatures provide integrity, unforgeability and non-repudiation properties for electronic documents, and play the important role in electronic commerce or modern cryptographic applications. A (t, n) threshold signature scheme [2] is a group oriented signature scheme, which allows any t or more signers to cooperatively sign messages on behalf of the group, but $t-1$ or fewer signers do not.

The concept of the public key cryptography was invented by Diffie and Hellman [3] solving some troublesome aspects of the key management in designing secure cryptosystems. In public key cryptosystems, each user chooses a pair of keys: private key and public key. Only the former is managed by the user secretly. The latter can be stored in a file managed by a system authority SA. It is unnecessary to safeguard the public key from exposure, since its secrecy is not required for secure communications or digital signatures. However, the public key cryptosystems suffer from the well-known authentication problem [5]. If an imposter supplies a valid but incorrect public key, a user could unknowingly encipher confidential data that would be decipherable by the imposter or be tricked into accepting messages with wrong signatures.

There are three possible approaches to provide authentication of public keys.

1) Certification-based public key. Each user chooses his pair of private/public keys, and registers him to a trusted third, certification authority SA. Then SA in turn issues a certificate corresponding to the public key of the user, which is the SA's signature on the pair of the user's public key and identity. SA should maintain a public key directory and a revocation list inquired by verifiers.

2) Identity-based public key [11]. Each user's public key is simply his identity string, and the corresponding private key is computed by SA with SA's private key. Verifiers need not check the public key. All users must trust SA, since SA knows their private keys.

3) Self-certified-based public key [5]. Each user chooses his pair of private/public keys. SA computes a certificate of user's public key and identity string with SA's private key. The verification of self-certified public key can be carried out in the subsequent cryptographic applications in a logically single step. Compared with Certification-based public key, Self-certified-based public key can reduce the computation and storage. Compared with Identity-based public key, Self-certified-based public key can provide more security confidence.

Girault defined the following three levels of trust in models:

1) The SA knows (or can easily compute) the users' private keys and is capable of impersonating any user without being detected.
2) The SA does not know the users' private keys, but it can still impersonate any user by generating a false certificate that may be used without being detected.
3) The SA does not know (and cannot compute) the users' private keys and if it generates false certificates for users, it can be proven.

With this definition, in schemes of level 1 and level 2, the SA must be fully trusted by the users, while in schemes of level 3, it is considered as a potentially powerful adversary that may use any means to impersonate the users.

Recently, Wu and Hsu [12] proposed a new (t, n) threshold signature scheme using self-certified public keys, in which the verification of public keys can be accomplished within the signature verification procedure. This signature scheme is more efficient since it is elaborated on the merits inherent in the self-certified public key system. However, the author of this paper points out that there are some problems in the Wu-Hsu scheme. The registration stage cannot work, since there is a deadlock in the computation of the self-certified individual public keys. Moreover, some t or more malicious signers can conspire together against the group, and the SA can also conspire with a malicious user to forge the group public key without being detected. Hence, The Wu-Hsu signature scheme at most is only of level 2.

Finally, we propose an improvement to counter the two attacks to achieve level 3. Its signature computation and verification are more efficient than those of the Wu-Hsu scheme. The system authority SA can identify the actual signers, while they are anonymous to outsiders.

2 Brief Review of the Wu-Hsu Threshold Signature Scheme Using Self-certified Public Keys

The Wu-Hsu scheme consists of four stages: the system setup, the registration, the individual signature generation and verification, and the group signature generation and verification. In the system environments, there exist a system authority SA and a clerk (CLK). The responsibilities of SA are to generate the system parameters and to issue users' individual public keys and the group public key, while those of CLK are to validate individual signatures and to combine them into a group signature.

2.1 System Setup

Initially, SA chooses a one-way hash function h, two large primes p and q, and a generator g of order q over $GF(p)$, where $q|(p-1)$. After that, SA determines a private-key/public-key pair (γ, β) where $\gamma \in Z_q^*$ and $\beta = g^\gamma \bmod p$. SA publishes p, q, g, h and β, while keeps γ secret.

2.2 Registration

Let $G = \{u_1, u_2, \cdots, u_n\}$ be the registering group of n users, ID_i be the identity information, such as name, address, etc., associated to $u_i \in G$ and GID be the identity information of G. The procedure for generating self-certified private-key/public-key pair of each group member $u_i \in G$ and the group G is described below.

Step 1. u_i randomly chooses a $(t-1)$-degree polynomial $f_i(z) = \sum_{j=0}^{t-1} a_{ij} z^j \bmod q$, where $a_{ij} \in Z_q$ for $j = 0, 1, \cdots, t-1$, and publishes the check vector $(cv_{i0} = g^{a_{i0}} \bmod p,\ cv_{i1} = g^{a_{i1}} \bmod p, \cdots, cv_{it-1} = g^{a_{it-1}} \bmod p)$.

Step 2. u_i computes $f_i(\mathrm{ID}j)$ and then sends it to $u_j \in G(j \neq i)$ via a secure channel.

Step 3. Upon receiving $f_j(\mathrm{ID}_i)$ sent from $u_j \in G(j \neq i)$, u_i checks its validity by the following equality:

$$g^{f_j(\mathrm{ID}_i)} = \prod_{k=0}^{t-1} cv_{jk}^{ID_i^k} \bmod p.$$

If it fails, he requests u_j to re-send$f_j(\mathrm{ID}_i)$.

Step 4. If all $f_j(\mathrm{ID}_i)$'s sent from other users are valid for $j = 1, 2, \cdots, n$ and $j \neq i$, u_i computes his secret shadow as $F_G(\mathrm{ID}_i) = \sum_{j=1}^{n} f_j(ID_i) \bmod q$.

Step 5. u_i computes the registering message (v_{i1}, v_{i2}) and sends it to SA, where

$$\begin{aligned} v_{i1} &= g^{-a_{i0}} \bmod p = cv_{i0}^{-1} \bmod p, \\ v_{i2} &= g^{-F_G(ID_i)} \bmod p. \end{aligned}$$

Step 6. Upon receiving all (v_{i1}, v_{i2})'s from $u_i \in G$ for $i = 1, 2, \cdots, n$, SA performs the following tasks:

(6-1) Checks the validity of v_{i1} and v_{i2} by the following equalities, respectively:

$$\begin{aligned} v_{i1} cv_{i0} &= 1 \bmod p, \\ v_{i2} &= \prod_{j=1}^{n} \prod_{k=0}^{t-1} cv_{jk}^{-ID_i^k} \bmod p. \end{aligned}$$

(6-2) Randomly chooses two $(t-1)$-degree polynomials $F_{\mathrm{SA}}(z)$ and $F_{\mathrm{GID}}(z)$ as

$$\begin{aligned} F_{\mathrm{SA}}(z) &= \sum_{j=1}^{t-1} b_j z^j + b_0 \bmod q, \quad \text{and} \\ F_{\mathrm{GID}}(z) &= \sum_{j=1}^{t-1} c_j z^j + h(Y_G || GID) \bmod q, \end{aligned}$$

where $b_0, b_j, c_j \in Z_q^*$(for $j = 1, 2, \cdots, t-1$).

(6-3) Computes the self-certified group public key Y_G for G as $Y_G = (\prod_{i=1}^{n} v_{i1}) g^{F_{\mathrm{SA}}(0)} - h(\mathrm{GID}) \bmod p$. It can be seen that $Y_G = g^{-F_G(0)} g^{F_{\mathrm{SA}}(0)} - h(\mathrm{GID}) \bmod p$.

(6-4) Computes the self-certified individual public key y_i for $u_i \in G (i = 1, 2, \cdots, n)$ as

$$y_i = v_{i2} g^{F_{\mathrm{SA}}(ID_i)} \beta^{d_i} - h(ID_i||GID) \bmod p,$$

where $d_i = F_{\mathrm{GID}}(\mathrm{ID}_i) - h(y_i||\mathrm{ID}_i||\mathrm{GID}) \bmod q$.

(6-5) Computes the witness w_i for $u_i \in G (i = 1, 2, \cdots, n)$ as $w_i = F_{\mathrm{SA}}(ID_i) + rF_{\mathrm{FID}} \bmod q$.

(6-6) Sends w_i to $u_i \in G$ and publish all self-certified individual public keys y_i's (for $i = 1, 2, \cdots, n$),and publish the group public key Y_G.

Step 7. After receiving w_i, u_i computes his self-certified private key x_i as $x_i = w_i - F_G(\mathrm{ID}_i) \bmod q$.

Step 8. Each user u_i validates y_i by checking the following equality:

$$g^{x_i} = (y_i + h(ID_i||GID)) \beta^{h(y_i||(ID_i||GID)} \bmod p.$$

Step 9. For verifying the self-certified group public key Y_G, each user u_i in G first computes and broadcasts $e_i = g^{x_i L_i} \bmod p$ to all other users in G, where $Li = \prod_{j=1, j\neq 1}^{t} \frac{-ID_j}{ID_i - ID_j} \bmod q$. Then each user can validate Y_G by checking the following equality:

$$\prod_{i=1}^{n} e_i = (Y_G + h(GID)) \beta^{h(Y_G||GID)} \bmod p.$$

It can be seen that

$$\prod_{i=1}^{n} e_i = g^{F_{SA}(0) - F_G(0) + rh(Y_G||GID)} \bmod p.$$

Hence, we denote the self-certified group private key of G as X_G, which is unknown to all users and SA, and $X_G = F_{\mathrm{SA}}(0) - F_G(0) + rh(Y_G||GID) \bmod q$.

2.3 Individual Signature Generation and Verification

Let M be the message to be signed. Without loss of generality, let $\mathrm{SG} = \{u_1, u_2, \cdots, u_t\}$ be the subgroup of G with t members who want to sign M on behalf of G. To generate the individual signature for M, each $u_i \in \mathrm{SG}$ performs the following steps:

Step 1. Chooses a random integer $k_i \in Z_q^*$ and computes $r_i = g^{k_i} \bmod p$. Then, u_i broadcasts r_i to other participating users $u_j \in \mathrm{SG}$ (for $j = 1, 2, \cdots, t$, and $j \neq i$) and CLK.

Step 2. Computes R after receiving all r_j's from $u_j \in \mathrm{SG}$ (for $j = 1, 2, \cdots, t$, and $j \neq i$) as $R = \prod_{i=1}^{t} r_i^{r_i} \bmod p$. Note that CLK also computes R by the same equality.

Step 3.Computes s_i as $s_i = k_i r_i h(M||R) + L_i x_i R \bmod q$, where $L_i = \prod_{j=1, j\neq i}^{t} \frac{-ID_j}{ID_i - IDj} \bmod q$.

Step 4. Sends (r_i, s_i) to CLK as the individual signature for M. Upon receiving (r_i, s_i) from $u_i \in SG$, CLK checks its validity and the authenticity of the self-certified individual public key y_i with respect to u_i by the following equality:

$$g^{s_i} = r_i^{r_i h(M||R)} ((y_i + h(ID_i||GID)) \beta^{h(y_i||ID_i||GID)})^{RL_i} \bmod p.$$

If it holds, then both (r_i, s_i) and y_i are valid.

2.4 Group Signature Generation and Verification

If all individual signatures are verified, then CLK computes $S = \sum_{i=1}^{t} S_i \bmod q$. Here, (R, S) is the group signature for M with respect to G. To verify the group signature, any verifier checks the following equality:

$$g^s = R^{h(M||R)} ((Y_G + h(GID)) \beta^{h(Y_G||GID)})^R \bmod p.$$

If it holds, then (R, S) is a valid group signature for M signed by G with respect to the self-certified public key Y_G. Note that only the group public key Y_G is involved in the verification equality, while self-certified individual public keys, $y_1, y_2, \cdots, y_t$, are excluded. Hence, the proposed scheme provides signer anonymity.

3 Security Analyses of the Wu-Hsu Threshold Signature Scheme

There are some problems in the Wu-Hsu threshold signature scheme. Some involve consistency, others involve security.

3.1 Computation of the Self-certified Individual Public Key

First the anonymous reviewers pointed the first error in *"(6-2) Randomly chooses two $(t-1)$-degree polynomials $F_{\mathrm{SA}}(z)$ and $F_{\mathrm{FID}}(z)$ as*

$$F_{\mathrm{SA}}(z) = \sum_{j=1}^{t-1} b_j z^j + b_0 \bmod q,$$

$$F_{\mathrm{FID}}(z) = \sum_{j=1}^{t-1} c_j z^j + h(Y_G||\mathrm{GID}) \bmod q,$$

where $b_0, b_j, c_j \in Z_q^*$ (for $j = 1, 2, \cdots, t-1$).

(6-3) Computes the self-certified group public key Y_G for G as $Y_G = (\prod_{j=1}^{n} v_{i1} g^{F_{\mathrm{SA}}(0)} - h(\mathrm{GID}) \bmod p$".

The computation of Y_G should be computed between $F_{\mathrm{SA}}(z)$ and $F_{\mathrm{FID}}(z)$, Since the computation of $F_{\mathrm{FID}}(z)$ needs Y_G.

However, there is a fatal error in the computation of the self-certified individual public key.

"(6-4) Computes the self-certified individual public key y_i for $u_i \in G$ $(i = 1, 2, \cdots, n)$ as

$$y_i = v_{i2} g^{F_{\mathrm{SA}}(ID_i)} \beta^{d_i} - h(\mathrm{ID}_i||\mathrm{GID}) \bmod p,$$

where $d_i = F_{\mathrm{GID}}(\mathrm{ID}_i) - h(y_i||\mathrm{ID}_i||\mathrm{GID}) \bmod q$".

There exists a deadlock in this equality, since the computation of y_i needs d_i, while that of d_i also needs y_i. There are perhaps two methods to solve this deadlock:

1) Modifying d_i:
If $d_i = F_{\mathrm{GID}}(\mathrm{ID}_i) - h(y_i||\mathrm{ID}_i||\mathrm{GID}) \bmod q$, SA could compute the self-certified individual public key y_i for $u_i \in G (i = 1, 2, \cdots, n)$. But this modification would results in a new security problem. Each user u_i would validate y_i by checking the following equality:

$$g^{x_i} = (y_i + h(ID_i||GID))\beta^{h(ID_i||GID)} \bmod p.$$

Any adversary can easily forge the pair (x_i, y_i) of the self-certified individual private/public keys without the private key γ of SA

2) Modifying $F_{\mathrm{SA}}(ID_i)$:
SA chooses randomly k_i and computes $k_i = g^{k_i} v_{i2} - h(ID_i||GID) \bmod p$. With the knowledge of its private key γ, SA can find c_i such that $v_{i2} g^{c_i} \beta^{d_i} = y_i + h(\mathrm{ID}_i||\mathrm{GID}) \bmod p$. Then SA chooses polynomial $F_{\mathrm{SA}}(z)$ with $F_{\mathrm{SA}}(\mathrm{ID}_i) = c_i \bmod q$, $i = 1, 2, \cdots, n$. If so, the polynomial $F_{\mathrm{SA}}(z)$ is at least of order $n - 1$. Hence all of the n users are required to cooperatively sign messages on behalf of the group G.

3.2 Verification of the Self-certified Group Public Key Y_G

"Step 9. For verifying the self-certified group key Y_G, each user u_i in G first computes and broadcasts $e_i = g^{x_i L_i} \bmod p$ to all other users in G, where $L_i = \prod_{j=1, j\neq 1}^{t} \frac{-ID_j}{ID_i - ID_j} \bmod q$. Then each user can validate Y_G by checking the following equality: $\prod_{i=1}^{n} e_i = (Y_G + h(GID))\beta^{h(Y_G||GID)} \bmod p$".

The order of the polynomials is $t - 1$, the polynomials can be reconstructed from the Lagrange interpolating polynomial given t secret shadows. Suppose that some t users want to verify the self-certified group key Y_G. Without loss of generality, these users are $u_1, u_2, \cdots, u_t$. Each user u_i first computes and broadcasts $e_i = g^{x_i L_i} \bmod p$ to all other users, where

$$L_i = \prod_{j=1, j\neq 1}^{t} \frac{-ID_j}{ID_i - ID_j} \bmod q.$$

Thus, the verification equality should be

$$\prod_{i=1}^{t} e_i = (Y_G + h(GID))\beta^{h(Y_G||GID)} \bmod p.$$

3.3 Coalition Attack by t Malicious Users

As Wu and Hsu pointed that the self-certified group private key of G is X_G, which is unknown to all users and SA, and

$$\begin{aligned} X_G &= F_{\mathrm{SA}}(0) - F_G(0) + \gamma h(Y_G||\mathrm{GID}) \bmod q \\ &= F_{\mathrm{SA}}(0) - F_G(0) + \gamma F_{GID}(0) \bmod q. \end{aligned}$$

However, the self-certified private key x_i of each user is the secret shadow of the X_G:

$$\begin{aligned} x_i &= w_i - F_G(\mathrm{ID}_i) \bmod q \\ &= F_{\mathrm{SA}}(\mathrm{ID}_i) - F_G(\mathrm{ID}_i) + \gamma F_{GID}(\mathrm{ID}_i) \bmod q. \end{aligned}$$

If t or more malicious users pool their secret shadows together, they can recover X_G by applying Lagrange interpolating polynomial [1]:

$$X_G = \sum_{i=1}^{t} x_i L_i \bmod q,$$

where $L_i = \prod_{j=1, j\neq i}^{t} \frac{-ID_j}{ID_i - ID_j} \bmod q$. Then each one of them can compute valid signatures alone on behalf of the group without the cooperation of other users afterwards. Obviously, this violates the intent of the group. This coalition attack is inherent in many threshold signature schemes [7] using threshold secret share scheme, as long as the private key can recover from secret shadows.

3.4 Coalition Attack by SA and a Malicious User

If SA computes other self-certified group key Y_G' as $Y'_G = g^k - h(\mathrm{GID}) \bmod p$, where k is random integer chosen by SA. He would compute

$$x = k + \gamma h(Y'_G||\mathrm{GID}) \bmod q,$$

such that

$$g_x = (Y'_G + h(\mathrm{GID}))\beta^{h(Y'_G||\mathrm{GID})} \bmod p.$$

With the knowledge of x, SA can easily compute the signature (R, S) of any message M such that

$$g^s = R^{h(M||R)}((Y'_G + h(GID))\beta^{h(Y'_G||\mathrm{GID})})^R \bmod p.$$

If Y'_G is published as the self-certified group public key, the outsiders cannot find this forgery. Suppose that some t users want to verify the self-certified group public key Y'_G. If one user u_i among the t users is the conspirer of the malicious SA, instead of computing $e_i = g^{x_i L_i} \bmod p$, he computes $e'_i = (Y'_G + h(GID))\beta^{h(Y'_G||\mathrm{GID})} g^{x_i L_i} / ((Y_G + h(GID))\beta^{h(Y_G||\mathrm{GID})}) \bmod p$. Then $(\prod_{j=1, j\neq i}^{t} e_j) e'_i = (Y'_G + h(GID))\beta^{h(Y'_G||\mathrm{GID})} \bmod p$. Hence the probability that the other users fail to find the forged group public key is non- negligible.

Though the system authority SA does not know the secret key of each user, he can still forge false group public key and then forge group signatures without being detected.

Therefore, the Wu-Hsu threshold signature scheme using self-certified public keys at most attains only level 2 defined by Girault.

4 Our Improvements

The system environments of the improved scheme are the same as those of the Wu-Hsu scheme.

4.1 System Setup

The system setup stage is the same as that of the original scheme.

Initially, SA chooses a one-way hash function h, two large primes p and q, and a generator g of order q over $GF(p)$, where $q|(p-1)$. After that, SA determines a private-key/public-key pair (γ, β) where $\gamma \in Z_q^*$ and $\beta = g^\gamma \bmod p$. SA publishes p, q, g, h, and β, while keeps γ secret.

4.2 Registration

Let $G = \{u_1, u_2, \cdots, u_n\}$ be the registering group of n users. Each user u_i with a pair of private/public keys (x_i, y_i) chooses a secondary private key a_i and computes secondary public key

$$\alpha_i = g^{a_i} \bmod p.$$

Then the user chooses a random integer w in Z_q and computes

$$\begin{aligned} r' &= g^w \bmod p, \\ s' &= w - (x_i y_i + a_i \alpha_i) h(r', \alpha_i, \mathrm{ID}_i) \bmod q, \end{aligned}$$

where ID_i is the identity information, such as name, address, etc., associated to $u_i \in G$. Then the user u_i sends $(r', \beta^w s', \beta^w \alpha_i, \beta_w \mathrm{ID}_i \bmod p)$ to the SA. The system authority SA computes $\beta^w = (r')^\gamma \bmod p$ and recovers $(s', \alpha_i, \mathrm{ID}_i)$. Then he verifies the secondary public key α_i by checking the following equality:

$$g^{s'} (y_i^{y_i} \alpha_i^{\alpha_i})^{h(r', \alpha_i, ID_i)} = r' \bmod p.$$

Note that the SA's ability of tracing the actual signers is dependent on the knowledge of the relationship between the secondary public keys and users' identities. Now the transmission of $(\alpha_i, \mathrm{ID}_i)$ is performed by using a secret way. Hence the identity of the secondary public key α_i is anonymous to anyone except for the user and SA.

Then SA determines GID, which includes the identity information of the group E, including the secondary public key (α_i for every user u_i, the threshold value t and the valid period. SA performs the following steps:

Step 1. SA randomly chooses an integer k and compute

$$r = g^k \bmod p.$$

Then SA computes

$$b_0 = k + \gamma h(GID, r) \bmod q.$$

Thus $g^{b_0} = \beta^{h(GID,r)} r \bmod p$. SA performs a verifiable secret sharing (VSS) scheme [8].

SA randomly chooses a $(t-1)$-degree polynomial $F_{\mathrm{GID}}(z)$ as

$$F_{\mathrm{GID}}(z) = \sum_{j=1}^{t-1} b_j z^j + b_0 \bmod q,$$

where $b_j \in Z_q^*$ for $j = 1, \cdots, t-1$, and publishes (GID, r) and the check vector $(cv_0, cv_1, \cdots, cv_{t-1})$:

$$\begin{aligned} cv_0 &= g^{b_0} = \beta^{h(\mathrm{GID},r)} r \bmod p, \\ cv_1 &= g^{b_i} \bmod p, \\ &\vdots \quad\quad \vdots \\ cv_{t-1} &= g^{b_{i-1}} \bmod p. \end{aligned}$$

Step 2. SA computes $w_i = F\mathrm{GID}(i) \bmod q$, and then sends (GID, r, w_i) to $u_i \in G$ via a secure channel similar to the way used by the user. SA also sends $(\mathrm{GID}, r, i, \alpha_i, v_i = g^{w_i} \bmod p)$ to CLK.

Step 3. Upon receiving w_i sent from SA, u_i checks its validity by the following equality:

$$g^{w_i} = \prod_{k=0}^{t-1} cv_k^{i^k} \bmod p.$$

If it fails, he requests SA to re-send w_i. CLK checks $(\mathrm{GID}, r, i, \alpha_i, v_i)$ by the following equality:

$$v_i = \prod_{k=0}^{t-1} cv_k^{i^k} \bmod p.$$

Here, CLK does not know the identity of the user u_i.

4.3 Individual Signature Generation and Verification

Let M be the message to be signed. Without loss of generality, let $\mathrm{SG} = \{u_1, u_2, \cdots, u_t\}$ be the subgroup of G with t members who want to sign M on behalf of G. To generate the individual signature for M, each signer $u_i \in$SG performs the following steps:

Step 1. Chooses a random integer $k_i \in Z_q^*$ and computes

$$r_i = g^{k_i} \bmod p.$$

Then, u_i broadcasts r_i to other participating signers $u_j \in$ SG (for $j = 1, 2, \cdots, t$, and $j \neq i$) and CLK.

Step 2. Computes R after receiving all r_j's from $u_j \in$ SG (for $j = 1, 2, \cdots, t$, and $j \neq i$) as

$$R = \prod_{i=1}^{t} r_i \bmod p.$$

Note that CLK also computes R by the same equality.

Step 3. Computes s_i as

$$s_i = k_i - (a_i h(\mathrm{GID}, r) + L_i w_i) E \bmod q,$$

where $E = h(M, R), L_i = \prod_{j=1, j\neq i}^{t} \frac{-j}{i-j} \bmod q$.

Step 4. Sends (i, s_i) to CLK as the individual partial signature for M.

4.4 Group Signature Generation and Verification

Upon receiving $s_i (i = 1, 2, \cdots, t)$, CLK computes $S = \sum_{i=1}^{t} s_i \bmod q$. Here, $(\mathrm{GID}, (1, 2, \cdots, t), r, E, S)$ is the group signature for M with respect to G. To verify the group signature, any verifier, including CLK, checks the following equality:

$$h(M, g^s((\alpha_1\alpha_2\cdots\alpha_t\beta)^{h(GID,r)}r)^E \bmod p) = E$$

If it holds, then $(\mathrm{GID}, (1, 2, \cdots, t), r, E, S)$ is a valid group signature for M signed by G. If CLK finds that the $(\mathrm{GID}, (1, 2, \cdots, t), r, E, S)$ is not valid, CLK checks the following equalities:

$$g^{s_i}(\alpha_i^{h(GID,r)}v_i^{L_i})^E = r_i \bmod p, \quad \text{for } i = 1, 2, \cdots, t,$$

where $E = h(M, R), L_i = \prod_{j=1, j\neq i}^{t} \frac{-j}{i-j} \bmod q$ with respect to the self-certified individual public key α_i and the secondary public key v_i of one anonymous signer u_i. If it holds, then both (r_i, s_i) and (α_i, v_i) are valid.

To reduce the storage of signatures, $(1, 2, \cdots, t)$ can be denoted by an integer, the ith bit of which is equal to 1 if and only if the secondary public key i is involved in the verification equality.

Note that the secondary public keys $\alpha_1, \alpha_2, \cdots, \alpha_t$ of the t signers are involved in the verification equality. Hence, the improvement provides traceability, by which system authority SA can identify the actual signers that are anonymous to outsiders.

Theorem 4.1 *If SA and users follows the protocol, the verification equalities are always to hold.*

Proof. First, $b_0 = k+\gamma h(GID, r) \bmod q$ directly implies $g^{b_0} = \beta^{h(GID,r)}r \bmod p$.

Second, $v_i = g^{w_i} = \prod_{k=0}^{t-1} cv_k^{i^k} \bmod p$. Then

$$s_i = k_i - (a_i h(\mathrm{GID}, r) + L_i w_i)E \bmod q$$

implies

$$g^{s_i}(g^{a_i h(GID,r)}g^{w_i L_i})^E = g^{k_i} \bmod p.$$

Hence, $g^{s_i}(\alpha_i^{h(GID,r)}(\prod_{k=0}^{t} cv_k^{i^k})^{L_i})^E = r_i \bmod p$. That is, $g^{s_i}(\alpha_i h(GID, r)v_i^{L_i})^E = r_i \bmod p$. In addition,

$$\prod_{i=1}^{t} g^{s_i}(\alpha_i^{h(GID,r)}v_i^{L_i})^E = \prod_{j=1}^{t} r_i \bmod p, \quad \text{for } i = 1, 2, \cdots, t,$$

implies

$$g^{\sum_{i=1}^{t} s_i}((\prod_{i=1}^{t} \alpha_i^{h(GID,r)})g^{\sum_{i=1}^{t} w_i L_i})^E = \prod_{i=1}^{t} r_i \bmod p.$$

Thus, $g^s((\alpha_1\alpha_2\cdots\alpha_t\beta)^{h(GID,r)}r)^E = R \bmod p$ by applying Lagrange interpolating polynomial. This is, $h(M, g^s((\alpha_1\alpha_2\cdots\alpha_t\beta)^{h(GID,r)}r)^E \bmod p) = E$. Therefore, three verification equalities are always to hold. *Q.E.D.*

5 Security Analyses and Performance

5.1 Security

We assume that one of the generalized ElGamal signature scheme is secure [6]. The signature (r, s) satisfies the following verification equality $g^s = y^m r^r \bmod p$. We also assume that the Schnorr signature scheme [10] is secure in the random oracle model [9].

First, The signature (b_0, r) satisfies $g^{b_0} = \beta^{h(GID,r)}r \bmod p$, which is the verification equality of a variant of the Schnorr signature scheme. Without the knowledge of the private key γ with $\beta = g^\gamma \bmod p$, anyone cannot compute this signature except SA. The group signature $(\mathrm{GID}, (1, 2, \cdots, t), r, E, S)$ satisfies $h(M, g^s((\alpha_1\alpha_2\cdots\alpha_t\beta)^{h(GID,r)}r)^E \bmod p) = E$ that can be regarded as the Schnorr signature for the message M with respect to the public key $Y = (\alpha_1\alpha_2\cdots\alpha_t\beta)^{h(GID,r)}r \bmod p$. If adversaries can forge signatures, they would solve the discrete logarithm. That is, they would find z with $Y = g^z \bmod p$. Hence the adversaries should be able to find $(z, r, \alpha_1, \alpha_2, \cdots, \alpha_t)$ such that $g^z = (\alpha_1, \alpha_2, \cdots, \alpha_t\beta)^{h(GID,r)}r \bmod p$.

SA is able to find $(z, r, 1, 2, \cdots, t)$ such that $g^z = (\alpha_1\alpha_2\cdots\alpha_t\beta)^{h(GID,r)}r \bmod p$. However, these secondary public keys are not the registered users' secondary public key. If SA generates false certificates for users, it can be proven.

Case 1: Suppose that the adversaries have an algorithm to solve this problem. When $t = 0$, they would find (z, r) such that $g^z = \beta^{h(GID,r)}r \bmod p$. Hence they would forge a Schnorr signature of the message GID.

Case 2: Assume that an insider attacker u_i could produce a private key $\hat{a}$ such that $g^{\hat{a}} = \alpha_1\alpha_2\cdots\alpha_t\beta \bmod p$, he would be able to compute the group signatures of messages. The insider attacker acquires all other secondary public keys β, $\alpha_1, \cdots, \alpha_{i-1}, \alpha_{i+1}, \cdots, \alpha_t$ of t innocent users. Then he chooses a random integer $\hat{a}$, computes $\hat{\alpha}_i = g^{\hat{a}} \bmod p$, $\alpha_i = \hat{\alpha}_i\beta^{-1}\prod_{j=1, j\neq i}^{t} \alpha_j^{-1} \bmod p$ and sends the quantity α_i as his secondary public key to SA. Thus $g^{\hat{a}} = \beta\prod_{i=1}^{t} \alpha_i \bmod p$. Finally he can generate valid signatures by using the false private key $\hat{a}$ and forged group signatures with identity GID.

In the improvement, however, the equality $g^{s'}(y_i^{y_i}\alpha_i^{\alpha_i})^{h(r', \alpha_i, ID_i)} = r' \bmod p$ can be regarded as the verification equality of the Schnorr signature of the message (α_i, ID_i) with respect to the public key $(y_i^{y_i}\alpha_i^{\alpha_i})$. Without the knowledge of the corresponding private key, anyone cannot compute Schnorr signature (s', r'). So the insider attacker u_i should know an integer u such that $g^u = (y_i^{y_i}\alpha_i^{\alpha_i}) \bmod p$. This equality can also be regarded as the verification equality of the ElGamal signature [4] of the message y_i with respect to the public key y_i. Thus the insider attacker u_i should know $(u - x_i y_i)\alpha_i^{-1} \bmod q$ such that $\alpha_i = g^{(u-x_i y_i)\alpha_i^{-1}} \bmod p$. Hence α_i is not a false secondary public key.

By the way, the public key β is not a false public key

either, since the users use it to encrypt their secondary public keys. Therefore the improvement can withstand the insider forgery attack.

If the system authority fails to verify if α_i is a false secondary public key, or the system authority conspires with the insider attacker, the insider attacker would obtain such a false secondary public key. If so, the insider attacker could compute valid group signatures without the cooperation of other signers.

However, other signers can detect such group signatures since they know the secondary public keys of the actual signers involved in the forged group signatures. They can defend themselves by asking the system authority to reveal (s', r', α_i, y_i) computed by each actual signer involved in the forged group signatures.

Case 3: Suppose that $t-1$ signers $u_1, u_2, \cdots, u_{t-1}$ and SA collude to forge signatures. If they have a way to compute (z, r) such that $g^z = (\alpha_1, \alpha_2, \cdots, \alpha_t\beta)^{h(GID,r)}r \bmod p$. With their secondary private keys of the corresponding secondary public keys $\alpha_1, \alpha_2, \cdots, \alpha_{t-1}, \beta$, they could computes (z', r) such that $g^{z'} = \alpha_t^{h(GID,r)}r \bmod p$. This is the Schnorr signature of message GID with respect the secondary public key α_t.

Case 4: Suppose that t signers $u_1, u_2, \cdots, u_t$ collude to forge signatures without registering. If they have a way to compute (z, r) such that $g^z = (\alpha_1, \alpha_2, \cdots, \alpha_t\beta)^{h(GID,r)}r \bmod p$. With their secondary private keys of the corresponding secondary public keys $\alpha_1, \alpha_2, \cdots, \alpha_t$, they could computes (z', r) such that $g^{z'} = \beta^{h(GID,r)}r \bmod p$. This is the Schnorr signature of message GID with respect to the public key β.

Therefore, the improved threshold signature scheme is secure under security assumption of the Schnorr signature and the ElGamal signture. The authentication of the self-certified secondary public keys can be confirmed simultaneously in the procedure of verifying group signature.

Case 5. There is a possible conspiracy attack as follows:
"Any t or more malicious signers can reveal their secret shares w_i's (i.e., $w_i = F_{GID}(i) \bmod q$) and then conspire together to obtain $b_0 = k + \gamma h(GID, r)$ by Lagrange interpolating polynomial. After that, these malicious signers can cooperate to generate a valid group signature as follows:

1) *Each signer randomly chooses a number k_i, computes $r_i = g^{k_i} \bmod p$, and broadcasts r_i.*

2) *Upon receiving all r_i's, each signer compute $\tilde{R} = \prod_{i=1}^{t} r_i \bmod p$.*

3) *Each signer computes $\tilde{E} = h(M, \tilde{R})$ and $\tilde{s}_i = k_i - (a_i h(GID, r) + b_0 t^{-1}) \bmod q$, and then broadcasts $\tilde{s}_i$.*

4) *Each signer computes $\tilde{S} = \sum_{i=1}^{t} \tilde{s}_i \bmod q$ and announces the generated group signature as $(GID, 1, 2, \cdots, t, \tilde{E}, \tilde{S})$.*

It is easy to see that the generated group signature will pass the following group signature verification: $h(M, g^{\tilde{S}}((\alpha_1 \cdots \alpha_t\beta)^{h(GID,r)}r)^{\tilde{E}} \bmod p = \tilde{E}$".

Though this valid signature of the message M is generated by a conspiracy, it does be generated by cooperation of at least t registered signers, rather than $t-1$ or fewer signers. Hence this signature does not violate the intent of the group. If they abuse the delegation, SA can detect them by using the relationship between the second public keys and the identities of these t malicious signers. In the Wu-Hsu threshold signature scheme, each one of the malicious signers can compute signatures alone on behalf of the group without the cooperation after the coalition. However, this possible attack does not endanger the improvement, it gives signers another way to cooperate to generate group signatures.

5.2 Performance of the Improvement

To generate a group signature, t modular exponentiations are required by t signers, while 3 modular exponentiations are required by CLK or a verifier to verify a group signature. This computation load of the improvement is slightly less than that of the Wu-Hsu scheme. Moreover, the registration load of the improvement is simpler than that of the Wu-Hsu scheme, though the storage of the signatures of the improvement is slightly more than that of the Wu-Hsu scheme. Contrary to the signer anonymity provided by the Wu-Hsu scheme, the improvement provides the system authority the ability to identify the actual signers of signatures since their secondary public keys are involved in the verification equality of signatures. They are anonymous to outsiders. Certainly, anonymity in my improvement is just obtained using pseudonymous since all the signatures produced by the same signer are linkable. However, unlinkability is the security requirement of group signatures.

6 Conclusions

We have pointed out that there are some problems in the Wu-Hsu scheme. The registration stage cannot work since there is a deadlock in the computation of the self-certified individual public keys. There is also a small error in the verification of the self-certified group key. Moreover, some t or more malicious signers can conspire together against the group, and the SA can also conspire with a malicious user to forge the group public key without being detected. Hence, the Wu-Hsu signature scheme using self certified public keys at most is only of level 2 defined by Girault. Finally, we propose an improvement to counter the two attacks to achieve level 3. We show that the improvement is secure under the security assumption that the Schnorr signature scheme and the generalized ElGamal signature scheme. The signature computation and verification of the improvement are more efficient than those of the Wu-Hsu scheme.

The main outstanding of the improvement is traceability, by which the system authority SA can identify the actual signers, though they are anonymous to outsiders.

Acknowledgment

The author would like to thank the anonymous reviewers for their valuable comments and suggestions that improve the presentation of this paper. This work is A Project Supported by Scientific Research Fund of Zhejiang Provincial Education Department.

References

[1] Dorothy E. R. Denning, *Cryptography and Data Security*. Massachusetts: Addison-Wesley, 1982.

[2] Y. Desmedt, "Society and group oriented cryptography: A new concept," in *Advances in Cryptology, CRYPTO'87*, LNCS 293, Springer-Verlag, pp. 120–127, 1987.

[3] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, no. 6, pp. 644–654, 1976.

[4] T. ElGamal, "A public-key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, vol. IT-31, pp. 469–472, July 1985.

[5] M. Girault, "Self-certified public keys," in *Advances in Cryptology, EUROCRYPT'91*, LNCS 547, Springer-Verlag, pp. 490–497, 1991.

[6] L. Harn, "Design of generalized elgamal type digital signature scheme based on discrete logarithm," *Electronics Letters*, vol. 31, no. 20, pp. 2025–2026, 1985.

[7] L. Harn, H.Y. Lin, and S. Yang, "Threshold cryptosystem with multiple secret sharing policies," *IEE Proc.-Comput. Digit. Tech*, vol. 141, no. 2, pp. 142–144, 1994.

[8] T. Pedersen, "Distributed provers with application to undeniable signatures," in *Proc. EUROCRYPT'91*, LNCS 547, Springer-Verlag, pp. 221–238, 1991.

[9] D. Pointcheval and J. Stern, "Security arguments for digital signatures and blind signatures," *Journal of Cryptology*, vol. 13, no. 3, pp. 361–396, 2000.

[10] C. P. Schnorr, "Efficient signature generation by smart cards," *Journal of Cryptology*, vol. 3, no. 3, pp. 161–174, 1991.

[11] A. Shamir, "Identity-based cryptosystem based on the discrete logarithm problem," in *Proc. CRYPTO'84*, pp. 47–53, 1984.

[12] T. S. Wu and C. L. Hsu, "Threshold signature scheme using self-certified public keys," *The Journal of Systems and Software*, vol. 67, pp. 89–97, 2003.

Zuhua Shao was born in Shanghai, People's Republic of China, on 30 April 1948. He received B.S. degree in mathematics and M.S. in algebra from the Northeastern Normal University, People's Republic of China in 1976 and 1981 respectively. Since 1990 he has taught computer science as an associated professor in the Hangzhou Institute of Financial Managers, The Industrial and Commerce Bank of China. Now he is a professor at the Zhejiang University of Science and Technology. His current research interests are cryptography and financial data security.