On the Security of Two Threshold Signature Schemes with Traceable Signers

Guilin Wang[1], Xiaoxi Han[1,2], and Bo Zhu[1]

[1] Infocomm Security Department
Institute for Infocomm Research
21 Heng Mui Keng Terrace, Singapore 119613
http://www.i2r.a-star.edu.sg/icsd/
{glwang,xiaoxi,zhubo}@i2r.a-star.edu.sg
[2] Laboratory of Computer Science, Institute of Software
Chinese Academy of Sciences, Beijing 100080, P.R. China

Abstract. A (t, n) threshold signature scheme allows t or more group members to generate signatures on behalf of a group with n members, while any $t-1$ or less members cannot do the same thing. In 2001, based on a variant of ElGamal digital signature scheme, Li et al. proposed two (t, n) threshold signature schemes with traceable signers. One of their schemes needs the assistance of a mutually trusted center, while the other does not. In this paper, we present a security analysis on their schemes. We first point out that in fact signers in their schemes are *untraceable*, since anybody can convert a valid threshold signature into a new one such that another subset of group members will be wrongly considered as the signers of the new threshold signature for the same message. Furthermore, we demonstrate an attack to show that their second threshold signature scheme is *insecure*. In our attack, $(n - t + 1)$ colluding members can control the group secret key. Therefore, they can generate valid threshold signature for any message without the help of other members. Furthermore, honest members cannot detect this security flaw in the system, since any t members can generate threshold signatures according to the prescribed protocols.

Keywords: digital signature, threshold signature, cryptanalysis.

1 Introduction

In 1987, Desmedt first introduced the concept of *group-orinted cryptography* [3]. In contrast to traditional cryptosystems, in a group-oriented cryptosystem, only several collaborative entities can perform a cryptographic operation (e.g. encryption, decryption, generation or verification of signatures.). *Threshold signature* is one such system, in which only t or more group members can generate signatures on behalf of a group with n members, while any $t-1$ or less members cannot do the same thing. On the other hand, to check the validity of a threshold signature, a verifier only needs to know the unique group public key. According to whether a verifier can trace back the signers of a threshold signature, there

J. Zhou, M. Yung, Y. Han (Eds.): ACNS 2003, LNCS 2846, pp. 111–122, 2003.

are two kinds of threshold signature schemes: with anonymous signers and with traceable signers. According to whether a trusted center is involved, threshold signature schemes can be classified into two types: with or without a trusted center. A scheme without the need of a trusted center is also called a distributed threshold signature scheme.

Based on RSA system, Desmedt and Frankel proposed the first threshold signature scheme in [4]. Gennaro et al. presented efficient threshold DSS signature schemes and threshold RSA signature schemes in [6] and [7] respectively. Stinson and Strobl proposed a provably secure distributed threshold Schnorr signature scheme and used it to design implicit certificates [12]. All these schemes do not provide the property to trace the identities of signers of a threshold signature.

In 2001, Li et al. proposed a variant of ElGamal digital signature scheme [5,9], and then based on this ElGamal type signature scheme, they constructed two (t, n) threshold signature schemes with traceable signers [10]: One of their schemes needs the assistance of a mutually trusted center, while the other does not. To show that their schemes are secure, they examined that several known attacks cannot be mounted in their schemes. However, we find that in fact their schemes cannot be used to trace the signers of a signature, and that their second scheme (i.e., the distributed one) is insecure.

More specifically, in this paper we first point out that the threshold signatures in their schemes are *untraceable* since anybody can convert a valid threshold signature into a new one such that another set of group members becomes the signers of the new threshold signature for the same message. Moreover, we demonstrate an attack to show that in their second threshold signature scheme, $(n - t + 1)$ colluding members can control the group secret key. Therefore, they can generate valid threshold signature for any message without the help of other members. Furthermore, honest members cannot detect any security flaw in the system, since under the assumption that the clerk is also corrupted, any t members can generate threshold signatures according to the prescribed protocols. In addition, we provide some countermeasures to prevent our attack.

The rest of this paper is organized as follows. Section 2 introduces Li et al.'s modified ElGamal signature scheme on which their threshold signature schemes are based. Section 3 reviews Li et al.'s second threshold signature scheme, and Section 4 presents our security analysis on their schemes. The conclusion is drawn in Section 5.

2 Modified ElGamal Signature Scheme

In this section, we review Li et al.'s modified ElGamal signature scheme [10] that is developed from generalized ElGamal digital signature schemes [9]. Li et al.'s threshold signature schemes are based on this modified ElGamal signature scheme.

System parameters:

- p: a large prime number such that $(p - 1)/2$ is also a prime number;
- g: a primitive element of the finite field $GF(p)$;

- $H(\cdot)$: a one-way hash function;
- x_u: the secret key of a user U;
- y_u: the related public key of the user U where $y_u = g^{x_u} \bmod p$.

Signature Generation. To sign a message m, a user U randomly chooses a number $k \in [1, p-1]$ and computes:

$$r = g^k \bmod p. \tag{1}$$

and then solves the following equation for s:

$$rs = (r + H(m))k + x_u \bmod p - 1. \tag{2}$$

The signature on the message m is (r, s).

Signature Verification. To verify a signature (r, s) on a message m, one checks whether the following equality holds:

$$g^{rs} \equiv r^{r+H(m)} y_u \bmod p. \tag{3}$$

3 Review of Li et al.'s Threshold Signature Schemes

In [10], Li et al. proposed two (t, n) threshold signature schemes with traceable signers: The first one needs a mutually trusted center while the second one does not. However, the methods of tracing back in these two schemes are the same. In this section we only review their second scheme which does not need a trusted center.

Their second scheme consists of seven parts: system parameter setup, group public key generation, individual signature generation, individual signature verification, threshold signature generation, threshold signature verification, and signer identification.

Part 1. System parameter setup

The following public system parameters are agreed by all group members.

- p, q: two large prime numbers such that $q|(p-1)$;
- g: a generator of order q in finite field $GF(p)$;
- $H(\cdot)$: a one-way hash function.

Part 2. Group public key generation

Let U_i, $i \in A = \{1, 2, \cdots, n\}$, be n group members in the system. For simplicity, we say A is the set of all group members, and similarly we denote a subset of group members by B, where $B \subset A$. To generate the group public key, each member U_i ($i \in A$) randomly selects his secret key x_i and a public identity ID_i. Then, he publishes his public key $y_i = g^{x_i} \bmod p$. When all y_i's ($i \in A$) are available, the group public key y is set by

$$y = \prod_{i \in A} y_i \bmod p.$$

Now, each group member uses Shamir's $(t, n-1)$ secret sharing scheme [11] to distribute his secret key to other $n-1$ members. That is, member U_i first selects a random $(t-1)$-degree polynomial $f_i(X)$ with $f_i(0) = x_i \bmod q$, sends the secret shadow $f_i(ID_j) \bmod q$ to each member U_j privately, and then publishes the related public key $y_{ij} = g^{f_i(ID_j)} \bmod p$ as public information, where $j \in A$ and $j \neq i$.

Part 3. Individual signature generation

When any t members of group A want to generate a signature for a message m, each member U_i, $i \in B$ ($B \subseteq A$ and $|B| = t$), selects a random integer $k_i \in [1, q-1]$, computes and broadcasts $r_i = g^{k_i} \bmod p$. After all r_i's are available, the following value R is calculated:

$$R = \prod_{i \in B} r_i \bmod p.$$

Using his secret key x_i, secret shadows $f_j(ID_i) \bmod q$ ($j \in A \setminus B$) and k_i, the group member U_i computes s_i from the following equation:

$$s_i R = (R + H(m))k_i + x_i + \sum_{j \in A \setminus B} f_j(ID_i) \cdot C_{Bi} \bmod q. \tag{4}$$

In the above equation, C_{Bi} is the Lagrange interpolating coefficient given by

$$C_{Bi} = \prod_{k \in B \setminus \{i\}} \frac{ID_k}{ID_k - ID_i} \bmod q. \tag{5}$$

Finally, each member U_i ($i \in B$) sends his individual signature (r_i, s_i) to a designated clerk.

Part 4. Individual signature verification

On receiving the individual signature (r_i, s_i) for message m from member U_i, the clerk uses public information y_i and y_{ij} ($j \in A \setminus B$) to verify the validity of (r_i, s_i) by

$$g^{s_i R} \equiv r_i^{R+H(m)} y_i \Big(\prod_{j \in A \setminus B} y_{ji} \Big)^{C_{Bi}} \bmod p. \tag{6}$$

If the above equation holds, the individual signature (r_i, s_i) from U_i is valid. At the same time, the clerk uses t pairs of public values (ID_i, y_i) ($i \in B$) to construct a Lagrange polynomial function $h(Y)$ as follows:

$$h(Y) = \sum_{i \in B} \Big(ID_i \prod_{j \in B \setminus \{i\}} \frac{Y - y_j}{y_i - y_j} \Big) \bmod q. \tag{7}$$

Later, the function $h(Y)$ will be used to trace the signers who signed the threshold signature for m.

Part 5. Threshold signature generation

After t individual signatures are received and validated, the threshold signature (R, S) for the message m is computed by

$$R = \prod_{i \in B} r_i \bmod p, \quad \text{and} \quad S = \sum_{i \in B} s_i \bmod q. \tag{8}$$

Part 6. Threshold signature verification

Any outsider can use the group public key y to check the validity of a threshold signature (R, S) for a message m by

$$g^{SR} \equiv R^{R+H(m)} y \bmod p. \tag{9}$$

If the above equation holds, the threshold signature (R, S) is valid. Otherwise, it is invalid.

Part 7. Signer identification

The authors of [10] implicitly assumed that the function $h(Y)$ is attached to the signature pair (R, S). Therefore, when a verifier wants to know the signers of a threshold signature (R, S) for a message m, he uses public values (ID_i, y_i) $(i \in A)$ to find all signers by the following equation:

$$ID_i \stackrel{?}{\equiv} h(y_i). \tag{10}$$

If the above equation holds, the group member U_i is one signer of the signature (R, S). Otherwise, he is not.

4 On the Security of Li et al.'s Schemes

In this section, we discuss the security of Li et al.'s threshold signature schemes. We first analyze the traceability of their two schemes and point out that in fact both of them are untraceable. Then, we demonstrate an attack on their second threshold signature scheme which does not need a trusted center. Finally, some improvements are also provided to prevent our attack.

4.1 On the Traceability of Li et al.'s Schemes

The methods of tracing back in their two threshold signature schemes are the same. That is, using public values (ID_i, y_i) and the function $h(Y)$ to determine whether Equation (10) holds. According to Equation (7), however, we know that the function $h(Y)$ is determined by public values, and that there is no intrinsic relationship between a signature pair (R, S) and $h(Y)$ since the same pair (R, S) for the message m may be generated by any t group members. Therefore, given a valid pair (R, S) for a message m, anybody can construct a function $h(Y)$ such that any t members, $U_{i_1}, \cdots, U_{i_t}$, could be the signers of (R, S). In addition, even if the values of ID_i and y_i are known only by group members, Li et al.'s

schemes are also untraceable since we have the following simple attack. Given two threshold signature pairs $(R, S, h(Y))$ and $(R', S', h'(Y))$ for two messages m and m', it is easy to see that $(R, S, h'(Y))$ and $(R', S', h(Y))$ are also two valid threshold signature pairs for messages m and m', respectively.

From the above discussion, it is clear that the signer tracing method proposed in [10] does not work. Thus, in Li et al.'s two schemes the signers of a threshold signature are untraceable, instead of traceable.

4.2 An Attack on Li et al.'s Second Scheme

In this subsection, we demonstrate an attack on Li et al.'s second threshold signature scheme in which n group members generate the group public key in a distributed way. We present the details about how $(n-t+1)$ colluding members can cheat other $(t-1)$ honest members by controlling the group secret key. In our attack, we make the following two assumptions:

- **Assumption 1.** Except $(t-1)$ honest members in the system, all other $(n-t+1)$ members are dishonest and collude together.
- **Assumption 2.** The designated clerk is also corrupted by dishonest members.

After our attack is presented, we will discuss why these two assumptions are necessary and why they are reasonable in applications. For simplicity, but without loss of generality, we assume that the first $(t-1)$ members, i.e., $U_1, \cdots, U_{t-1}$, are honest members and all other $(n-t+1)$ members are dishonest. We further assume that as the head of dishonest members, U_n controls the group secret key, while other colluding members tolerate his faults in the group public key generation.

It is obvious that, using the group secret key, anyone of these $(n-t+1)$ malicious members can forge a valid threshold signature on any message independently. Furthermore, we demonstrate that if $n \in B$ or $B = \{1, 2, \cdots, t-1, j\}$ where $j \in \{t, \cdots, n-1\}$, then under the help of the corrupted clerk, the t members coming from B can generate valid individual signatures and threshold signatures. Moreover, even in the case where $|\bar{B} \cap \{t, t+1, \cdots, n-1\}| \geq 2$, the t members in $\bar{B}$ also can generate valid threshold signatures but one malicious member in $\bar{B}$ cannot generate valid individual signatures. For example, if $t = 8$ and $n = 10$, we know that in a secure threshold signature scheme, a valid threshold signature can only be generated by 8 or more group members. However, in Li et al.'s second scheme our attack enables members U_8, U_9 and U_{10} to control the group secret key x and to cheat other seven honest members. By using the known group secret key x, any of U_8, U_9 or U_{10} can independently generate valid threshold signature for any message. At the same time, under the help of the corrupted clerk, any eight out of ten members can generate valid threshold signatures. Furthermore, except in the case where the signers consist of U_8, U_9 and six out of seven honest members, colluding members also can provide valid individual signatures.

We emphasize that the above property is important in real world, since it allows dishonest members to cheat honest members that the system is normal

and secure until a forged threshold signature on a message appears. Otherwise, even if U_n controls the group secret key but any t members cannot generate a threshold signature by the prescribed protocols, honest members will soon doubt the security and usability of the system.

The details of our attack are given as follows. The whole procedure consists of three steps: member U_n controlling the group private key, member U_n distributing secret shares, and dishonest members generating valid individual signatures.

Step 1. Member U_n controlling the group private key

In the group public key generation of Li et al.'s second scheme, it is not required that all public keys y_i's should be published simultaneously. Thus, member U_n can be the last one to publish his public key y_n. By choosing a random number x as the group secret key, he first sets the group public key by $y = g^x \bmod p$. Then, when all other y_i's ($i \in \{1, \cdots, n-1\}$) are published, he computes and publishes his public key y_n as follows:

$$y_n = y \cdot \prod_{i=1}^{n-1} y_i^{-1} \bmod p.$$

Hence, all members in group A will take y as the group public key, since the following equation holds:

$$y = y_n \cdot \prod_{i=1}^{n-1} y_i = g^x \bmod p.$$

Therefore, member U_n has controlled the group private key x corresponding to y. Of course, member U_n does not know his private key x_n corresponding to y_n since he cannot find the discrete logarithm of y_n to the base g.

Step 2. Member U_n distributing secret shares

The difficulty is how member U_n can distribute his secret key x_n to other members even though he does not know the value of x_n. For this sake, U_n does as follows.

1. Firstly, U_n assumes that he has chosen a $(t-1)$-degree polynomial $f_n(X)$ such that $f_n(0) = x_n$, where x_n is the unknown but fixed number satisfying $y_n = g^{x_n} \bmod p$. At the same time, he selects $(t-1)$ random numbers $b_i \in [0, q-1]$, and sets $f_n(ID_i) = b_i$, for each $i \in \{1, 2, \cdots, t-1\}$.
2. Secondly, he sends $f_n(ID_i)$ privately to each member U_i, $i \in \{1, 2, \cdots, t-1\}$. However, U_n cannot send $f_n(ID_j)$ to each of his conspirators since he does not know the value of $f_n(ID_j)$ for each $j \in \{t, t+1, \cdots, n-1\}$. But, as U_n's conspirators, each member U_j tolerates this fault.
3. Thirdly, U_n has to publish the related public key y_{nj}, for all $j \in \{1, 2, \cdots, n-1\}$. Of course, for $i \in \{1, 2, \cdots, t-1\}$, U_n computes y_{ni} by $y_{ni} = g^{b_i} \bmod p$. Moreover, we now explain that U_n can also carry out $y_{nj} = g^{f_n(ID_j)}$ for each $j \in \{t, t+1, \cdots, n-1\}$ even though he does not know the value of

$f_n(ID_j)$. According to the Lagrange interpolating formula, the following equation holds:

$$y_n = g^{f_n(0)} = g^{C_{B_j j} \cdot f_n(ID_j)} \cdot \prod_{i=1}^{t-1} g^{C_{B_j i} \cdot f_n(ID_i)} \bmod p, \quad \forall\, j \in \{t, t+1, \cdots, n-1\}.$$

In the above equation, $B_j = \{1, 2, \cdots, t-1, j\}$, $C_{B_j j}$ is the Lagrange interpolating coefficient given by Equation (5), and $f_n(ID_i) = b_i$. Therefore, U_n can get the value of $y_{nj} = g^{f_n(ID_j)} \bmod p$ by the following equation:

$$y_{nj} = (y_n \cdot \prod_{i=1}^{t-1} g^{-C_{B_j i} \cdot b_i})^{C_{B_j j}^{-1}} \bmod p, \quad \forall j \in \{t, t+1, \cdots, n-1\}. \tag{11}$$

After all y_{nj} ($j \in \{1, 2, \cdots, n-1\}$) are computed, U_n publishes them. Any member can verify that all y_{nj}'s are consistent since the following equation holds:

$$y_n \equiv \prod_{j \in B} y_{nj}^{C_{Bj}} \bmod p, \quad \forall B \subseteq \{1, 2, \cdots, n-1\} \text{ and } |B| = t. \tag{12}$$

When U_n and all other members published all y_{ij}, $i, j \in \{1, 2, \cdots, n\}$ and $i \neq j$, the system is set up. After that, if necessary, any dishonest member can use the known group secret key x to forge valid threshold signature on any message m. That is, he first chooses a random $k \in [0, q-1]$ and computes $R = g^k \bmod p$. Then, he gets S from equation $RS = (R + H(m))k + x \bmod p$. It is easy to know that such forged pair (R, S) is a valid threshold signature on message m, since it satisfies Equation (9). Furthermore, as we have mentioned, under the help of the corrupted clerk, dishonest members can also generate valid individual signature. So they can cheat honest members that the system is normal and secure.

Step 3. Dishonest members generating their individual signatures

Now, we assume t members of a subset B want to sign a message m. According to the individual signature generation Equation (4), U_n cannot generate valid individual signature since he does not know the value of x_n. In addition, if $n \notin B$, any malicious member U_j ($t \leq j \leq n-1$) cannot generate valid individual signature since he does not know the value of $f_n(ID_j)$. However, note that if $n \in B$, any U_j ($t \leq j \leq n-1$) can generate his individual signature. Therefore, under our assumption 2, we will see that the corrupted clerk can help dishonest members to generate valid individual signatures in two cases: (1) $n \in B$ and (2) $B = \{1, 2, \cdots, t-1, j\}$ where $j \in \{t, t+1, \cdots, n-1\}$. In the following, we only describe how dishonest members can generate their valid individual signatures in case 1. As we mentioned above, in this case $n \in B$, any other (honest or dishonest) member can generate his individual signature normally. Therefore, we now focus on how member U_n can generate his individual signature.

In Li et al.'s scheme, it is also not required that all r_i's should be published simultaneously, thereby member U_n can be the last one to publish r_n. That is, U_n first selects a random number $k \in [0, q-1]$, and sets

$$R = g^k \bmod p. \tag{13}$$

When all other r_i's have been broadcast, he computes and broadcasts the following r_n:

$$r_n = R \cdot \prod_{i \in B/\{n\}} r_i^{-1} \bmod p.$$

Consequently, each member U_j ($j \in B/\{n\}$) computes R by $R = r_n \cdot \prod_{i \in B/\{n\}} r_i \bmod p$ ($= g^k \bmod p$). Then, by using Equation (4), each U_j generates and sends his individual signature (r_i, s_i) to the clerk. The corrupted clerk reveals the values of all (r_i, s_i)'s to U_n. To get his individual signature on the message m, U_n first solves S from the following equation

$$SR = (R + H(m))k + x \bmod q. \tag{14}$$

Next, he computes his individual signature (r_n, s_n) as follows.

$$r_n = R \prod_{i \in B \setminus \{n\}} r_i^{-1} \bmod q, \quad \text{and} \quad s_n = S - \sum_{i \in B \setminus \{n\}} s_i \bmod q. \tag{15}$$

Finally, U_n sends (r_n, s_n) to the clerk so that the clerk can generate the threshold signature (R, S) for the message m. If necessary, the clerk publishes all individual signatures (r_i, s_i) ($i \in B$) as the evidences that all members in B indeed generated valid individual signatures for the message m. After all (r_i, s_i)'s have been broadcast, each member in B can verify the validity of each pair (r_i, s_i) by using Equation (6). Up to this piont, U_n generated his individual signature pair (r_n, s_n).

The following theorem proves that the above (R, S) is a valid threshold signature for the message m, and that (r_n, s_n) is U_n's valid individual signature for the message m.

Theorem 1. *The above procedure that U_n generates his individual signature is successful. That is,*

(1) The pair (R, S) generated by Equation (14) is a valid threshold signature for the message m, and
(2) The pair (r_n, s_n) generated by Equation (15) is U_n's valid individual signature for the same message m.

Proof: (1) It is obvious that the pair (R, S) generated by Equation (14) satisfies Equation (9). We now prove (2): i.e., we need to show that the pair (r_n, s_n) generated by Equation (15) satisfies Equation (6). This is justified by the following equalities.

$$\begin{aligned}
g^{s_n R} &= g^{(S - \sum_{i \in B/\{n\}} s_i)R} \bmod p \\
&= g^{(R+H(m))k} \cdot y \cdot \left[\prod_{i \in B/\{n\}} \left(r_i^{R+H(m)} \cdot y_i \cdot (\prod_{j \in A/B} y_{ji})^{C_{Bi}} \right) \right]^{-1} \bmod p \\
&= \left(R \prod_{i \in B/\{n\}} r_i^{-1} \right)^{R+H(m)} \cdot y \prod_{i \in B/\{n\}} y_i^{-1} \cdot (\prod_{j \in A/B} \prod_{i \in B/\{n\}} y_{ji}^{-C_{Bi}}) \\
&= r_n^{R+H(m)} \cdot y \prod_{i \in B/\{n\}} y_i^{-1} \cdot \prod_{j \in A/B} y_{jn}^{C_{Bn}} \cdot \prod_{j \in A/B} (\prod_{i \in B} y_{ji}^{-C_{Bi}}) \bmod p \\
&= r_n^{R+H(m)} \cdot y \prod_{i \in B/\{n\}} y_i^{-1} \cdot \prod_{j \in A/B} y_{jn}^{C_{Bn}} \cdot \prod_{j \in A/B} y_j^{-1} \bmod p \\
&= r_n^{R+H(m)} \cdot y_n \cdot (\prod_{j \in A/B} y_{jn})^{C_{Bn}} \bmod p.
\end{aligned}$$

When any U_j ($j \in \{t, t+1, \cdots, n-1\}$) and $t-1$ honest members want to generate a threshold signature, U_j can generate his valid individual signature analogously, and similar result as Theorem 1 can also be proved.

4.3 Remarks and Improvements

In this subsection, we give some remarks on our attack and the two assumptions listed in the beginning of Section 4.2. In addition, several improvements on Li et al.'s second scheme are provided to prevent the above attack.

We first comment that our attack is stronger in the sense that it allows malicious members to provide valid individual signatures. Actually, in the Li et al.'s original second scheme, only the clerk checks the validity of individual signatures. In this case, t members in any subset B can generate valid threshold signature as follows. A malicious member U_j in B first controls the value of R as in the foregoing attack. He then solves S from equation $SR = (R + H(m))k + x \bmod q$, and sends the threshold signature pair (R, S) to the corrupted clerk.

Now, we point out that the success of our attack does not depend on whether the number of dishonest members $(n-t+1)$ is less than t. However, of course, our attack has practical meanings only in the case where $(n-t+1) < t$, since the basic assumption of all threshold cryptosystems is that there are at most $(t-1)$ malicious members. Therefore, our assumption 1, $(n-t+1)$ members colluding together, is reasonable in application settings.

We remark that our assumption 2 is also reasonable in many applications. In Li et al.'s second scheme, the clerk only performs some computations that anyone can do, and the computational result, i.e. the threshold signature (R, S), is publicly verifiable. In a practical system, it is unlikely to set the clerk as a trusted party. Therefore, dishonest members can corrupt and make use of him for a long time, since the clerk is a designated entity. Furthermore, a natural invariant of Li et al.' second scheme is to remove the clerk by requiring each member to broadcast his individual signature (r_i, s_i). In this case, we do not need assumption 2 any more. At the same time, dishonest members can generate their individual signatures by taking the same steps as in section 4.2.

From the foregoing description, we know that our attack results from the insecurity of the group public key generation protocol in Li et al.'s second scheme. Therefore, to avoid our attack, each member should publish his public key y_i simultaneously. One simple way is to require that all members first commit their y_i's and then open their commitments to reveal y_i's. Another choice is to require that each member proves his knowledge of the discrete logarithm of y_i to the base g by using interactive or non-interactive knowledge proof protocols [1,2]. The third choice is to use Gennaro et al.'s distributed key generation protocol for discrete-log cryptosystems [8] as the group public key generation protocol. At the same time, we know that in our attack dishonest members can use other members' r_i's and s_i's to generate their counterparts. Therefore, all members should first commit them before revealing their values.

5 Conclusion

In this paper, we presented a security analysis to Li et al.'s two (n, t) threshold signature schemes with traceable signers [10]. We first pointed out that in their schemes the signers of a signature are in fact untraceable, since anybody can convert a valid threshold signature into a new one for the same message. As a result of it, another subset of group members will be wrongly identified as the signers of the new threshold signature for the same message. Furthermore, on their scheme without a mutually trusted center, we demonstrated an attack that allows $(n - t + 1)$ colluding members to control the group secret key and then generate valid threshold signature for any message. However, honest members cannot detect this security flaw in the system since any t members can generate threshold signatures according to the specified protocols. Consequently, colluding dishonest members can cheat honest members successfully. In addition, some countermeasures are provided to prevent our attack.

References

1. J. Camenisch and M. Stadler. Effient group signature schemes for large groups. In: *Advances in Cryptology – CRYPTO'97, LNCS 1294*, pp. 410–424. Springer-Verlag, 1997.
2. D. Chaum and T.P. Pedersen. Wallet databases with observers. In: *Advances in Cryptology – CRYPTO'92, LNCS 740*, pp. 89–105. Springer-Verlag, 1993.
3. Y. Desmedt. Society and group oriented cryptography: A new concept. In: *Advances in Cryptology – CRYPTO'87, LNCS 293*, pp. 120–127. Springer-Verlag, 1988.
4. Y. Desmedt and Y. Frankel. Shared generation of authenticators and signatures (Extended Abstract). In: *Advances in Cryptology – CRYPTO'91, LNCS 576*, pp.457–469. Springer-Verlag, 1991.
5. T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 1985, 31: 469–472.
6. R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Robust threshold DSS signatures. In: *Advances in Cryptology-EUROCRYPT'96, LNCS 1070*, pp. 354–371. Springer-Verlag, 1996. Also appears in *Information and Computation*, 2001, 164(1): 54–84.
7. R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Robust and Efficient Sharing of RSA Functions. In: *Advances in Cryptology – CRYPTO'96, LNCS 1109*, pp. 157–172. Springer-Verlag, 1996. Also appears in *Journal of Cryptology*, 2000, 13: 273–300.
8. R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Secure distributed key generation for discrete-log based cryptosystems. In *Advances in Cryptology - EUROCRYPT'99, LNCS 1592*, pp. 295–310. Springer-Verlag, 1999.
9. L. Harn and Y. Xu. Design of generalized ElGamal type digital signature schemes based on discrete logarithm. *Electronic Letters*, 1994, 24(31): 2025–2026.
10. Z.-C. Li, J.-M. Zhang, J. Luo, W. Song, and Y.-Q. Dai. Group-oriented (t, n) threshold digital signature schemes with traceable signers. In: *Topics in Electronic Commerce* (*ISEC 2001*), *LNCS 2040*, pp. 57–69. Springer-Verlag, 2001.

11. A. Shamir. How to share a secret. *Communications of the ACM*, 1979, 22(11): 612–613.
12. D.R. Stinson and R. Strobl. Provably secure distributed Schnorr signatures and a (t, n) threshold scheme for implicit certificates. In: *Information Security and Privacy* (*ACISP'01*), *LNCS 2119*, pp. 417–434. Springer-Verlag, 2001.