*Int. J. Communications, Network and System Sciences*, 2014, 7, 22-29
Published Online January 2014 (http://www.scirp.org/journal/ijcns)
http://dx.doi.org/10.4236/ijcns.2014.71003

# Asynchronous Secret Reconstruction and Its Application to the Threshold Cryptography

**Lein Harn[1], Changlu Lin[2*]**
[1]Department of Computer Science Electrical Engineering, University of Missouri-Kansas City, Kansas City, USA
[2]School of Mathematics and Computer Sciences, Fujian Normal University, Fuzhou, China
Email: *cllin@fjnu.edu.cn

Received October 26, 2013; revised November 26, 2013; accepted December 3, 2013



## ABSTRACT

**In Shamir's $(t,n)$ threshold of the secret sharing scheme, a secret $s$ is divided into $n$ shares by a dealer and is shared among $n$ shareholders in such a way that (a) the secret can be reconstructed when there are $t$ or more than $t$ shares; and (b) the secret cannot be obtained when there are fewer than $t$ shares. In the secret reconstruction, participating users can be either legitimate shareholders or attackers. Shamir's scheme only considers the situation when all participating users are legitimate shareholders. In this paper, we show that when there are more than $t$ users participating and shares are released asynchronously in the secret reconstruction, an attacker can always release his share last. In such a way, after knowing $t$ valid shares of legitimate shareholders, the attacker can obtain the secret and therefore, can successfully impersonate to be a legitimate shareholder without being detected. We propose a simple modification of Shamir's scheme to fix this security problem. Threshold cryptography is a research of group-oriented applications based on the secret sharing scheme. We show that a similar security problem also exists in threshold cryptographic applications. We propose a modified scheme to fix this security problem as well.**

## KEYWORDS

**Shamir's $(t,n)$ Secret Sharing Scheme; Secret Reconstruction; Threshold Cryptography; Threshold Decryption; Asynchronous Networks**

## 1. Introduction

Secret sharing schemes (SSs) were first introduced by both Blakley [1] and Shamir [2] separately in 1979 as a solution for safeguarding cryptographic keys and have been studied extensively in the literature. SS has become one of the most basic tools in cryptographic research. In Shamir's $(t,n)$ SS, a secret, $s$, is divided into $n$ shares by a dealer and shares are sent to shareholders secretly. The security requirements of a $(t,n)$ SS satisfy that (a) the secret can be reconstructed when there are $t$ or more than $t$ shares; and (b) the secret cannot be obtained when there are fewer than $t$ shares. Shamir's $(t,n)$ SS is based on the linear polynomial and is unconditionally secure. There are other types of SS. For example, Blakely's scheme [1] is based on the geometry, Mignotte's scheme [3] and Asmuth-Bloom's scheme [4] are based on the Chinese Remainder Theorem (CRT).

In the secret reconstruction, participating users can be either legitimate shareholders or attackers. Shamir's scheme only considers the situation when all participating users are legitimate shareholders. When there are more than $t$ users participating and shares are released asynchronously in the secret reconstruction, an attacker can always release his share last. In such a way, after knowing $t$ valid shares, the attacker can obtain the secret and therefore, can successfully impersonate to be legitimate shareholder without being detected. One simple way to overcome this security problem is to authenticate every

*Corresponding author.

participating user to be a legitimate sharesholder before reconstructing the secret. Since all user authentication schemes are one-to-one type of interactions between one *prover* and one *verifier*, this approach may slow down the secret reconstruction significantly especially when there are a large number of users who participated in the process.

In this paper, we propose a simple modification of Shamir's scheme to fix the security problem in the secret reconstruction. Our solution does not use any user authentication. The secret can be reconstructed successfully only when all participating users are legitimate shareholders and release their shares honestly. If there are any attackers among participating users, the secret cannot be reconstructed by users since the attacker does not own any valid share. Furthermore, the attacker cannot obtain the secret from partially released valid shares of legitimate shareholders.

In Shamir's $(t,n)$ SS, every share can only be used for one time to recover the secret. This is because once the secret has been recovered, then all shares are released and the secret is no longer secret. Therefore, the $(t,n)$ SS is not very efficient. To improve its efficiency, the $(t,n)$ SS has been incorporated with public-key cryptography in the threshold cryptography. The group-oriented threshold cryptosystem was first introduced by Desmedt [5] in 1987. In such a system, each group, instead of each individual group member, publishes a single group public key. The corresponding private key of the group's public key is divided into *n* shares and is shared among *n* group members following a $(t,n)$ SS [1-4], where *t* is a predefined threshold value. Threshold cryptography is the study of efficient multiparty computation protocols for cryptographic functions (e.g. signing or decrypting), in which each group member has a share of the private key which allows the computation of such function. Threshold cryptography utilizes some computational assumptions, such as factoring a composite integer or solving the discrete logarithm, to enable shares of group members to be reused for multiple times. Similar to the $(t,n)$ SS, participating users in a threshold cryptographic application can be either legitimate group members or attackers. All threshold cryptographic applications only consider the situation when all participating users are legitimate group members. When there are more than $t$ users participating and values of users are released asynchronously in a threshold application, an attacker can always release his computed values last. In such a way, after knowing $t$ valid values of legitimate group members, the attacker can obtain the valid output of the cryptographic function and therefore, can successfully impersonate to be a legitimate group member without being detected. We also propose a modified scheme to fix this security problem.

***Related Works.*** The security of cryptographic schemes/protocols can be classified into two types, computational security and unconditional security. Computational security assumes that the adversary has bounded computing power that limits the adversary to solving hard mathematical problem, such as factoring a large composite integer into two primes. Unconditional security means that the security holds even if the adversary has unbounded computing power. Research on developing cryptographic schemes/protocols with unconditional security has received wide attention recently. Shamir's $(t,n)$ SS scheme is based on a linear polynomial and is unconditionally secure.

Shamir's $(t,n)$ SS is very simple; but if the secret reconstruction is performed over networks, possible threats make the secret reconstruction very complicate. In fact, attackers who do not own valid shares may impersonate to be shareholders who participated in the secret reconstruction. In 1985, Chor *et al*. [6] proposed the notion of verifiable secret sharing (VSS). VSS enables shareholders to verify that their shares are valid without revealing their shares. There are vast research papers on VSS [6-8] in the literature. VSS is a complicate process which requires additional information and processing time.

How the secret should be reconstructed fairly is another research problem. When all other participating shareholders honestly present their shares in the secret reconstruction process, a dishonest shareholder can always exclusively get the secret by presenting a fake share and thus the others get nothing but a fake secret. Although protocols have been developed to detect fake shares [9-12], they do not prevent a dishonest shareholder from gaining this advantage. Even if the cheater is detected, this problem still persists as the cheater has already obtained the secret. The first protocol to solve this problem is proposed by Tompa *et al*. [10]. Most fair secret reconstruction proposals share one basic idea that utilizes a process where information is revealed slowly [13]. Chaum *et al*. [14], and Beaver *et al*. [15] considered the general problem of fair multiparty computation. Most of these works are based on a computational model, *i.e*., some computational assumptions are used like the existence of an oblivious transfer protocol or the quadratic residuosity assumption. In 1995, Lin *et al*. [16] proposed the first fair secret reconstruction protocol in which the real secret can be reconstructed as a whole entity without simultaneously releasing constraint. Cheating immune SSs through which the cheaters gain no advantage over honest participants by submitting invalid shares, are proposed in [17,18]; but secret and shares are limited to be either binary or from $GF(p)$.

Our proposed scheme is not a VSS since in our scheme, shares are the only information of shareholders

to prevent attackers from obtaining the secret and shares are not protected in the secret reconstruction. Furthermore, our proposed scheme cannot prevent a dishonest shareholder from presenting a fake share last in the secret reconstruction. In such a way, the dishonest shareholder can always exclusively get the secret but others get nothing but a fake secret. How the secret should be reconstructed fairly is a different research problem.

In a threshold signature scheme [19], when there are *t* or more than *t* group members, a group signature can be generated successfully. The group signature can be verified by any verifier using the group public key. On the other hand, in a threshold decryption scheme [20,21], any sender of a secret message can generate a cipher-text to the group using the group public key. When there are *t* or more than *t* group members, the group cipher-text can be decrypted successfully. All existing threshold cryptographic algorithms [19-26] only consider the situation when all participating users are legitimate group members. In this paper, we point out that when there are more than *t* participating users and computed valued are released asynchronously in a threshold application, an attacker can obtain the valid output of the application and therefore, can impersonate to be a legitimate group member without being detected.

We summarize the contributions of this paper in the following.

- We point out a security problem in Shamir's $(t,n)$ SS when there are more than *t* participating users and shares are released asynchronously in the secret reconstruction.
- A modified $(t,n)$ SS based on Shamir's $(t,n)$ is proposed to fix the security problem.
- We point out a similar security problem in all existing threshold algorithms when there are more than *t* participating users in threshold applications.
- A modified threshold decryption is proposed to fix the security problem.

**The rest of this paper is organized as follows.** In Section 2, we review Shamir's $(t,n)$ SS scheme and point out a security problem in Shamir's $(t,n)$ SS. In Section 3, we present a modified SS to fix the security problem when there are more than *t* participating users and shares are released asynchronously in the secret reconstruction. In Section 4, we point out the similar security problem and propose a solution to fix the security problem of a threshold decryption scheme. We conclude in Section 5.

## 2. Review of Shamir's (*t*, *n*) SS [2]

In Shamir's (*t*, *n*) SS based on a linear polynomial, the dealer *D* is responsible to select a secret and generate shares of the secret to *n* shareholders, $U=\{U_1,U_2,\cdots,U_n\}$. The scheme consists of two algorithms as illustrated in **Figure 1**.

Shamir's SS satisfies security requirements of the (*t*, *n*) SS, that are, (a) the secret can be reconstructed with *t* or more than *t* shares; and (b) no information about the secret can be obtained with fewer than *t* shares. In other words, if there are exactly *t* legitimate shareholders participated in the secret reconstruction, Shamir's scheme can recover the secret. Shamir's secret reconstruction scheme can be generalized to take more than *t* shares. For example, if there are *j* (*i.e.*, $t \le j \le n$) participated shareholders with their shares, $\{f(x_1), f(x_2),\cdots, f(x_j)\}$, in the secret reconstruction, the secret can be recovered as follows.

$$s = f(0) = \sum_{r=1}^{j} f(x_r) \prod_{v=1, v\neq r}^{j} \frac{-x_v}{x_r - x_v} \bmod p.$$

During secret reconstruction, participated users can be either legitimate shareholders or attackers. Shamir's scheme only considers the situation when all participated users are legitimate shareholders. When there are more than *t* users participated in the secret reconstruction and shares are released asynchronously, an attacker can always release his share last. After knowing *t* valid shares of legitimate shareholders, since the secret polynomial, $f(x)$, having degree $t-1$, the attacker can reconstruct the secret. Furthermore, the attacker can successfully forge a valid share on the polynomial, $f(x)$, without being detected. Thus, Shamir's (*t*, *n*) SS is no longer secure if there are more than *t* users participated in the secret reconstruction.

## 3. Proposed (*t*, *n*) Secret Sharing Scheme

In this section, we propose a (*t*, *n*) SS to fix the security problem of Shamir's (*t*, *n*) SS when there are more than *t* participated users and shares are released asynchronously

***Share generation***

Dealer *D* picks a random polynomial *f*(*x*) of degree $t-1$: $f(x)=a_0+a_1x+\cdots+a_{t-1}x^{t-1} \bmod p$, such that the secret is $s=f(0)=a_0$, and coefficients, $a_i, i=0,1,\cdots,t-1$, are in $GF(p)$, with $p > s$ and $p$ is a prime. *D* computes *n* shares, $y_r=f(x_r), r=1,2,\cdots,n$, where $x_r$ is the public information associated with shareholder, $U_r$. Then, the dealer distributes each share, $y_r$, to corresponding shareholder $U_r$ secretly.

***Secret reconstruction***

Assume that *t* shareholders, $\{U_1,U_2,\cdots,U_t\}$, work jointly to recover the secret, *s*. Shareholders release their shares and use the Lagrange interpolating formula,

$s=f(0)=\sum_{l=1}^{t} f(x_l)\prod_{v=1,v\neq l}^{t}\frac{-x_v}{x_l-x_v} \bmod p$, to recover the secret.

**Figure 1. Shamir's (*t*, *n*) SS.**

in the secret reconstruction. The outcome of our proposed secret reconstruction is either (a) the secret if all participated users are legitimate shareholders; or (b) not the secret if there are attackers. The basic idea is that the dealer in Shamir's (*t*, *n*) SS scheme selects $k$ (*i.e.*, $kt > n-1$, for example, if $t=2$, $n=5$, then $k=3$. We will prove this condition in ***Theorem* 1**) random polynomials, $f_l(x), l=1,2,\cdots,k$, having degree $t-1$ each, and generates shares, $f_l(x_r), l=1,2,\cdots,k$, for each shareholder, $U_r$, where $x_r$ is the public information of shareholder, $U_r$. For the secret, *s*, the dealer can always find integers, $w_l, d_l, l=1,2,\cdots,k$, in $GF(p)$, such that $s=\sum_{l=1}^{k} d_l f_l(w_l)$, where $w_i \neq w_j$, for every pair of $i$ and $j$, and $w_i \notin \{x_1, x_2, \cdots, x_n\}$. The dealer makes these integers, $w_l, d_l, l=1,2,\cdots,k$, publicly known.

We assume that there are $j$ (*i.e.*, $t \le j \le n$) participated shareholders, $\{U_1, U_2, \cdots, U_j\}$, in the secret reconstruction. Each shareholder $U_r$ uses his shares, $f_l(x_r), l=1,2,\cdots,k$, to compute and release one Lagrange component, $c_r = \sum_{l=1}^{k} d_l f_l(x_r) \prod_{v=1, v\neq r}^{j} \frac{w_l - x_v}{x_r - x_v} \bmod p$, to other participants. After knowing $c_r, r=1,2,\cdots,j$, each shareholder can recover the secret as $s=\sum_{r=1}^{j} c_r \bmod p$. We outline this scheme, ***Scheme* 1**, in **Figure 2**.

***Share generation***

The dealer *D* selects $k$ (*i.e.*, $kt > v-1$) random polynomials, $f_l(x), l=1,2,\cdots,k$, having degree $t-1$ each and generates shares, $f_l(x_r), l=1,2,\cdots,k$, for each shareholder, $U_r$, where $x_i$ is the public information for shareholder, $U_i$. For the secret, *s*, the dealer can always find integers, $w_l, d_l, l=1,2,\cdots,k$, in $GF(p)$, such that $s=\sum_{l=1}^{k} d_l f_l(w_l)$, where $w_i \neq w_j$ for every pair of $i$ and $j$, and $w_i \notin \{x_1, x_2, \cdots, x_n\}$. The dealer makes these integers, $w_l, d_l, l=1,2,\cdots,k$, publicly known. The dealer generates shares, $f_l(x_r), l=1,2,\cdots,k$, of shareholder, $U_r$.

***Secret reconstruction***

We assume that there are $j$ (*i.e.*, $t \le j \le n$) shareholders, $\{U_1, U_2, \cdots, U_j\}$, participated in the secret reconstruction.

1) Each shareholder $U_r$ uses his shares, $f_l(x_r), l=1,2,\cdots,k$, to compute and release one Lagrange component, $c_r = \sum_{l=1}^{k} d_l f_l(x_r) \prod_{v=1, v\neq r}^{j} \frac{w_l - x_v}{x_r - x_v} \bmod p$, to other shareholders secretly.

2) After knowing $c_r, r=1,2,\cdots,j$, each shareholder computes $s=\sum_{r=1}^{j} c_r \bmod p$.

**Figure 2. Scheme 1—Proposed (*t*, *n*) SS.**

**Theorem 1.** *The outcome of Scheme* 1 *is either* (*a*) *the secret when all participated users are legitimate shareholders*; *or* (*b*) *not the secret when there are attackers.*

***Proof.*** In Scheme 1, if all participated users are legitimate shareholders and act honestly to compute their Lagrange components, $c_r$, in Step 1, then in Step 2, we get

$$\sum_{r=1}^{j} c_r \bmod p = \sum_{r=1}^{j}\sum_{l=1}^{k} d_l f_l(x_r) \prod_{v=1, v\neq r}^{j} \frac{w_l - x_v}{x_r - x_v} \bmod p$$

$$= \sum_{l=1}^{k}\sum_{r=1}^{j} d_l f_l(x_r) \prod_{v=1, v\neq r}^{j} \frac{w_l - x_v}{x_r - x_v} \bmod p = \sum_{l=1}^{k} d_l f_l(w_l) = s.$$

The outcome is the secret.

On the other hand, if there are attackers in the secret reconstruction, since attackers do not know any valid shares, the outcome of Scheme 1 is not the secret.

In the following discussion, we want to determine whether attackers can still recover the secret from partially released Lagrange components, $c_r$, of legitimate shareholders. We analyze the security of the scenario which gives an attacker the most information to recover the secret. We assume that there are *n* users participated in the secret reconstruction and among them, there are $n-1$ legitimate shareholders and the attacker is the last one to release his Lagrange component. Since each released Lagrange component is a linear function of *kt* coefficients of polynomials, $f_l(x), l=1,2,\cdots,k$, having degree $t-1$, the attacker can obtain $n-1$ Lagrange components to form $n-1$ equations. The condition, $kt > n-1$ (*i.e.*, *kt* is the number of unknown coefficients of polynomials, $f_l(x), l=1,2,\cdots,k$, having degree $t-1$ each), prevents the attacker solving the secret polynomials, $f_l(x), l=1,2,\cdots,k$. Thus, the attacker cannot recover the secret in Scheme 1. We have come to this conclusion without making any computational assumption. Thus, the proposed scheme is unconditionally secure.■

***Remark* 1.** *For the secret, s, the dealer needs to select* $w_i \neq w_j$, *for every pair of i and j and the secret is* $s=\sum_{l=1}^{k} d_l f_l(w_l)$. *If* $w = w_i = w_j$, *for every pair of i and j, the attacker can still recover the secret after knowing t partially released Lagrange components,* $c_r$, *of legitimate shareholders. This is because in this case, the secret,* $s=\sum_{l=1}^{k} d_l f_l(w)$, *is a share of the additive sum of polynomials,* $\sum_{l=1}^{k} d_l f_l(x)$, *having degree* $t-1$. *Each participated shareholder,* $U_i$ *needs to use his shares to compute and release the Lagrange component,* $c_r = \sum_{l=1}^{k} d_l f_l(x_r) \left( \prod_{v=1, v\neq r}^{j} \frac{w - x_v}{x_r - x_v} \right) \bmod p$. *The attacker can*

*recover the additive sum of shares,* $\sum_{l=1}^{k} d_l f_l(x_r)$, *from each released Lagrange component* $c_i$. *Thus, after knowing t additive sum of shares, the attacker can recover secret as,*

$$\sum_{l=1}^{k} d_l f_l(x_r)\left(\prod_{v=1,v\neq r}^{t}\frac{w-x_v}{x_r-x_v}\right)\bmod p$$
$$=\sum_{l=1}^{k} d_l f_l(w)\bmod p = s.$$

## 4. Proposed Threshold Decryption Scheme

In this section, we present two $(t,n)$ threshold decryption schemes; one is a basic scheme and the other one is a modified scheme, in which the security of both schemes is based on the computational difficulty of solving the discrete logarithm problem. We use the basic scheme to point the security problem when there are more than *t* users participated in a threshold application. The threshold decryption scheme is one of the group-oriented threshold cryptographic algorithms. In this application, a group manager (GM), instead of each individual group member, publishes a single group public key. The corresponding private key of the group's public key is divided into *n* shares and is shared among *n* group members. Then, any *t* or more than *t* group members can enable a threshold application.

### 4.1. Basic Scheme

We assume that there are $n$ members, $U_i$, for $i=1,2,\cdots,n$, forming a group.

***Initialization***

The GM selects two large public primes, $p$ and $q$, such that $q$ divides $p-1$, $GF(q)$ is a unique subgroup of $GF(p)$ with order $q$, and one public generator, $g$, from $GF(q)$. GM selects a private key, $s\in GF(q)$, and computes the public key of the group, $y=g^s \bmod p$. GM acts like a dealer in Shamir's $(t,n)$ SS to select a random polynomial $f(x)$ of degree $t-1$: $f_i(x)=a_0+a_1x+\cdots+a_{t-1}x^{t-1} \bmod q$, such that the private key is $s=f(0)=a_0$, and all coefficients, $a_i$, for $i=0,1,\cdots,t-1$, are in $GF(q)$ with $q>s$. GM computes $n$ shares, $f(x_r), r=1,2,\cdots,n$, where $x_r$ is the public information associated with group member $U_r$. Each share, $f(x_r)$, is sent to group member, $U_r$, secretly.

***Generation of a cipher-text***

We adopt the ElGamal's encryption scheme [27]. With access to the group public key, $y$, the sender can generate a cipher-text, $(c_1,c_2)$, of message *m* by computing $c_1=g^k \bmod p$, where $k$ is a random integer in $GF(q)$, and $c_2=m\cdot K \bmod p$, where $K=y^k \bmod p$. The sender sends the cipher-text, $(c_1,c_2)$, to the group.

***Decryption of a cipher-text***

Let us assume that $j$ (*i.e.*, $t\le j\le n$) group members, $\{U_1,U_2,\cdots,U_j\}$, work together to decrypt a cipher-text. Each group member, $U_i$, uses his private share, $f(x_i)$, to compute a partial session key, $s_i=c_1^{f(x_i)\prod_{r=1,r\neq i}^{j}\frac{-x_r}{x_i-x_r}\bmod q} \bmod p$. The value, $s_i$, is sent to other group members participated in the decryption. After collecting all partial session keys, the session key, $K$, can be computed as

$$K=\prod_{r=1}^{j} s_i = g^{k\left(\sum_{i=1}^{j} f(x_i)\prod_{r=1,r\neq i}^{j}\frac{-x_r}{x_i-x_r}\right)\bmod q} = g^{ks}=y^k \bmod p.$$

Each participated group member can decrypt the cipher-text as $m=c_2\cdot K^{-1} \bmod p$.

***Security discussion***

In a practical application, participated users in a threshold decryption can be either legitimate group members or attackers. We assume that an attacker has collected $t$ valid partial session keys,

$$s_i=c_1^{f(x_i)\prod_{r=1,r\neq i}^{j}\frac{-x_r}{x_i-x_r}\bmod q} \bmod p,\ i=1,2,\cdots,t,$$

of legitimate group members. Then, the attacker can compute

$$s_i'=s_i^{\left(\prod_{r=1,r\neq i}^{j}\frac{-x_r}{x_i-x_r}\right)^{-1}\left(\prod_{r=1,r\neq i}^{t}\frac{-x_r}{x_i-x_r}\right)\bmod q}$$
$$=c_1^{f(x_i)\prod_{r=1,r\neq i}^{t}\frac{-x_r}{x_i-x_r}\bmod q} \bmod p,$$
$$i=1,2,\cdots,t.$$

The real session key, $K$, can be obtained by computing

$$K=\prod_{i=1}^{t} s_i' = g^{k\left(\sum_{i=1}^{t} f(x_i)\prod_{r=1,r\neq i}^{t}\frac{-x_r}{x_i-x_r}\right)\bmod q} = g^{ks}=y^k \bmod p.$$

Thus, the attacker can decrypt the cipher-text as $m=c_2\cdot K^{-1} \bmod p$. Furthermore, the attacker can successfully forge a valid partial session key of other legitimate group member, say $U_{t+1}$, as

$$\prod_{i=1}^{t} s_i^{\left(\prod_{r=1,r\neq i}^{j}\frac{-x_r}{x_i-x_r}\right)^{-1}\left(\prod_{r=1,r\neq i}^{t}\frac{x_{t+1}-x_r}{x_i-x_r}\right)\prod_{r=1,r\neq t+1}^{j}\frac{-x_r}{x_i-x_r}\bmod q}$$
$$=g^{k\left(\sum_{i=1}^{t} f(x_i)\prod_{r=1,r\neq i}^{t}\frac{x_{t+1}-x_r}{x_i-x_r}\right)\prod_{r=1,r\neq t+1}^{j}\frac{-x_r}{x_i-x_r}\bmod q}$$
$$=c_1^{f(x_{t+1})\prod_{r=1,r\neq t+1}^{j}\frac{-x_r}{x_i-x_r}\bmod q} \bmod p$$
$$=s_{t+1},$$

without being detected in this process. The security problem of this basic scheme is caused by the fact that the modular exponentiation of each share, $c_1^{f(x_i)} \bmod p$, can be obtained from the partial session key, $s_i$. With

any $t$ modular exponentiations of shares, the attacker can successfully recover the modular exponentiation of the constant term of the polynomial and to recover the session key, $K$. In the following subsection, we propose a simple modification to fix this security problem.

## 4.2. Modified Scheme

***Initialization***

GM selects two large public primes, $p$ and $q$, such that $q$ divides $p-1$, $GF(q)$ is a unique subgroup of $GF(p)$ with order $q$, and one public generator, $g$, from $GF(q)$. GM selects a private key, $s \in GF(q)$, and computes the public key of the group, $y = g^s \bmod p$. GM acts like a dealer in Shamir's $(t,n)$ SS to select two random polynomials, $f(x)$ and $g(x)$, having degree $t-1$ each: $f(x) = a_0 + a_1 x + \cdots + a_{t-1}x^{t-1} \bmod q$ and $g(x) = b_0 + b_1 x + \cdots + b_{t-1}x^{t-1} \bmod q$, such that the private key, $s$, is divided into $s = a + b \bmod q$, with $f(0) = a_0 = a$, and $g(1) = b$, all coefficients, $a_i$ and $b_i$ for $i = 0,1,\cdots,t-1$, are in $GF(q)$, with $q > a,b$. GM computes a pair of shares, $f(x_i)$ and $g(x_i)$, for each shareholder, $U_i$, where $x_r$ is the public information associated with $U_r$. Each pair of shares, $f(x_i)$ and $g(x_i)$, is sent to group member, $U_i$, secretly.

***Generation of a cipher-text***

This part of process is the same as the basic scheme. A group cipher-text, $(c_1, c_2)$, of message $m$ is generated by computing $c_1 = g^k \bmod p$ and $c_2 = m \cdot K \bmod p$, where $k$ is a random integer from $GF(q)$ and $K = y^k \bmod p$. The sender sends the cipher-text, $(c_1, c_2)$, to the group.

***Decryption of cipher-text***

Let us assume that $j$ (*i.e.*, $t \le j \le n$) group members, $\{U_1, U_2, \cdots, U_j\}$, work together to decrypt the cipher-text. Each group member, $U_i$, uses his pair of private shares, $f(x_i)$ and $g(x_i)$, to compute a partial session key, $s_i = c_1^{f(x_i)\prod_{r=1,r\ne i}^{j}\frac{-x_r}{x_i-x_r} + g(x_i)\prod_{r=1,r\ne i}^{j}\frac{1-x_r}{x_i-x_r} \bmod q} \bmod p$. The value $s_i$ is sent to other participated shareholders. After collecting all partial session keys from other group members, the member can decrypt the cipher-text as $m = c_2 \cdot K^{-1} \bmod p$. We outline this scheme, ***Scheme* 2**, in **Figure 3**.

**Theorem 2.** *In Scheme* 2*, if all participated users are legitimate group members and act honestly, the cipher-text,* $(c_1, c_2)$*, of the message m can be decrypted successfully.*

***Proof.*** If all group members act honestly, the real session key, $K$, can be obtained as

$$\prod_{r=1}^{j} s_i = g^{k\left(\left(\sum_{i=1}^{j} f(x_i)\prod_{r=1,r\ne i}^{j}\frac{-x_r}{x_i-x_r}\right)+\left(\sum_{i=1}^{j} g(x_i)\prod_{r=1,r\ne i}^{j}\frac{1-x_r}{x_i-x_r}\right)\right) \bmod q}$$

$$= g^{k(f(0)+g(1))} = g^{ks} = y^k \bmod p = K.$$

***Initialization***

GM selects two large public primes, $p$ and $q$, such that $q$ divides $p-1$, $GF(q)$ is a unique subgroup of $GF(p)$ with order $q$, and one public generator, $g$, from $GF(q)$. GM selects a private key, $s \in GF(q)$, and computes the public key of the group, $y = g^s \bmod p$. GM selects two random polynomials, $f(x)$ and $g(x)$, having degree $t-1$ each: $f(x) = a_0 + a_1 x + \cdots + a_{t-1}x^{t-1} \bmod q$ and $g(x) = b_0 + b_1 x + \cdots + b_{t-1}x^{t-1} \bmod q$, such that the private key $s$ is divided into $s = a + b \bmod q$, with $f(0) = a_0 = a$, and $g(1) = b$, all coefficients, $a_i$ and $b_i$ for $i = 0,1,\cdots,t-1$, are in $GF(q)$ with $q > a,b$. GM computes shares, $f(x_i)$ and $g(x_i)$, for each shareholder, $U_i$, where $x_r$ is the public information associated with $U_r$.

***Generation of cipher-text***

A group cipher-text, $(c_1, c_2)$, of message m is generated by computing $c_1 = g^k \bmod p$ and $c_2 = m \cdot K \bmod p$, where $k$ is a random integer from $GF(q)$ and $K = y^k \bmod p$. He sender sends the cipher-text, $(c_1, c_2)$, to the group.

***Cipher-text decryption***

Let us assume that $j$ (*i.e.*, $t \le j \le n$) group members, $\{U_1, U_2, \cdots, U_j\}$, work together to decrypt the cipher-text.

1) Each group member, $U_i$, uses his pair of private shares, $f(x_i)$ and $g(x_i)$, to compute a partial session key, $s_i = c_1^{f(x_i)\prod_{r=1,r\ne i}^{j}\frac{-x_r}{x_i-x_r} + g(x_i)\prod_{r=1,r\ne i}^{j}\frac{1-x_r}{x_i-x_r} \bmod q} \bmod p$. The value $s_i$ is sent to other participated group members.

2) After collecting all partial session keys from other group members, the session key is computed as $K = \prod_{i=1}^{j} s_i \bmod p$.

Each participated member can decrypt the cipher-text as $m = c_2 \cdot K^{-1} \bmod p$.

**Figure 3.** ***Scheme* 2—Proposed (*t, n*) threshold decryption scheme.**

The session key can be used to recover the message as $m = c_2 \cdot K^{-1} \bmod p$. ■

***Security discussion***

In this modified scheme, the partial session key is $s_i = c_1^{f(x_i)\prod_{r=1,r\ne i}^{j}\frac{-x_r}{x_i-x_r} + g(x_i)\prod_{r=1,r\ne i}^{j}\frac{1-x_r}{x_i-x_r} \bmod q} \bmod p$. Since the exponent of each partial session key is a linear combination of two shares, $f(x_i)$ and $g(x_i)$, attackers cannot separate these two shares to obtain the modular exponentiation of each share. In other words, attackers need to collect all partial session keys to be able to decrypt the cipher-text of the group. Therefore, if there are attackers participated in the decryption, the cipher-text cannot be decrypted.

We want to determine whether attackers can still decrypt the group cipher-text from a portion of partial ses-

sion keys which are released from legitimate group members. There are $2t$ coefficients in polynomials, $f(x_i)$ and $g(x_i)$, having degree $t-1$ each. If an attacker can obtain $v'$ (*i.e.*, $v' > 2t$) additive sum of shares, $f(x_i)\prod_{r=1,r\neq i}^{j}\frac{-x_r}{x_i-x_r}+g(x_i)\prod_{r=1,r\neq i}^{j}\frac{1-x_r}{x_i-x_r} \bmod q$, from released partial session keys, the attacker is able to solve both polynomials, $f(x)$ and $g(x)$. However, this attack is computational infeasible due to the difficulty of solving the discrete logarithm problem.

***Remark* 3.** *For any secret, s, the dealer needs to select two different points on the polynomials, $f(x)$ and $g(x)$, for example, $f(0)$ and $g(1)$. If two points are the same such as $s=a+b=f(0)+g(0)$, the secret, s, is a share of the additive sum of polynomials, $f(x)+g(x)$, having degree $t-1$. The partial session key of $U_i$ is $s_i = c_1^{(f(x_i)+g(x_i))\prod_{r=1,r\neq i}^{j}\frac{-x_r}{x_i-x_r} \bmod q} \bmod p$. If one attacker has obtained $t$ partial session keys, $s_i = c_1^{(f(x_i)+g(x_i))\prod_{r=1,r\neq i}^{j}\frac{-x_r}{x_i-x_r} \bmod q} \bmod p$, $i=1,2,\cdots,t$, of legitimate group members, the attacker can compute*

$$s_i' = s_i^{\left(\prod_{r=1,r\neq i}^{j}\frac{-x_r}{x_i-x_r}\right)^{-1}\left(\prod_{r=1,r\neq i}^{t}\frac{-x_r}{x_i-x_r}\right) \bmod q}$$

$$= c_1^{(f(x_i)+g(x_i))\prod_{r=1,r\neq i}^{t}\frac{-x_r}{x_i-x_r} \bmod q} \bmod p, \quad i=1,2,\cdots,t.$$

*Therefore, the session key, $K$, can be obtained by computing*

$$\prod_{i=1}^{t} s_i' = g^{k\left(\sum_{i=1}^{t}(f(x_i)+g(x_i))\prod_{r=1,r\neq i}^{t}\frac{-x_r}{x_i-x_r}\right) \bmod q}$$

$$= g^{ks} = y^k \bmod p = K.$$

## 5. Conclusion

We pointed out security problems of a (*t*, *n*) SS and a threshold algorithm. The security problems occurred when there are more than *t* participated users and shares/values that are released asynchronously in the secret reconstruction/threshold application. Since all existing networks are asynchronous networks and we cannot exclude the probability that with more than *t* users participating in a secret reconstruction/threshold application, our paper has made significant contributions to addressing the security problems and proposing solutions. We believe that we have opened a new research direction in both the secret sharing and the threshold cryptography.

## Acknowledgements

This research is supported by the National Natural Science Foundations of China under Grant No. 61103247.

## REFERENCES

[1] G. R. Blakley, "Safeguarding Cryptographic Keys," *Proceedings of Americian Federation of Information Processing Societies* (*AFIPS*'79) *National Computer Conference*, 25-28 February 1979, California, pp. 313-317.

[2] A. Shamir, "How to Share a Secret," *Academic Common Market*, Vol. 22, No. 11, 1979, pp. 612-613. http://dx.doi.org/10.1145/359168.359176

[3] M. Mignotte, "How to Share a Secret," *Cryptography-Proceedings of the Workshop on Cryptography*, Burg Feuerstein, 29 March-2 April 1982, pp. 371-375.

[4] C. A. Asmuth and J. Bloom, "A Modular Approach to Key Safeguarding," *IEEE Transactions on Information Theory*, Vol. IT-29, No. 2, 1983, pp. 208-210. http://dx.doi.org/10.1109/TIT.1983.1056651

[5] Y. Desmedtm, "Society and Group Oriented Cryptography: An New Concept," *Advances in Cryptography—7th Annual International Cryptology Conference* (*CRYPTO* '87), Santa Barbara, 16-20 August 1987, pp. 120-127.

[6] B. Chor, S. Goldwasser, S. Micali and B. Awerbuch, "Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults," *Proceedings of the 26th IEEE Symposium on the Foundations of Computer Science*, Portland, 21-23 October 1985, pp. 383-395.

[7] M. H. Dehkordi and S. Mashhad, "New Efficient and Practical Verifiable Multi-SSs," *Information Sciences*, Vol. 178, No. 9, 2008, pp. 2262-2274. http://dx.doi.org/10.1016/j.ins.2007.11.031

[8] J. C. Benaloh, "Secret Sharing Homomorphisms: Keeping Shares of a Secret Secret," *Advances in Cryptology—6th Annual International Cryptology Conference* (*CRYPTO* '86), Santa Barbara, 17-21 August 1987, pp. 251-260.

[9] E. F. Brickle and D. R. Stinson, "The Detection of Cheaters in Threshold Schemes," *Advances in Cryptology—9th Annual International Cryptology Conference* (*CRYPTO* '88), Santa Barbara, 21-25 August 1988, pp. 564-577.

[10] M. Tompa and H. Woll, "How to Share a Secret with Cheaters," *Journal of Cryptology*, Vol. 1, No. 3, 1988, pp. 133-138. http://dx.doi.org/10.1007/BF02252871

[11] T. Rabin and M. Ben-Or, "Verifiable Secret Sharing and Multiparty Protocols with Honest Majority," *Proceedings of the 21st ACM Symposium on the Theory of Computing*, Seattle, Washington DC, 14-17 May 1989, pp. 73-85.

[12] D. Chaum, C. Crepeau and I. Damgard, "Multipary Unconditionally Secure Protocols," *Proceedings of the 20th ACM Symposium on the Theory of Computing*, Chicago, 2-4 May 1988, pp. 11-19.

[13] J. He and E. Dawson, "Shared Secret Reconstruction," *Designs, Codes and Cryptography*, Vol. 14, No. 3, 1998, pp. 221-237. http://dx.doi.org/10.1023/A:1008200702849

[14] D. Chaum, I. Damgard and J. van de Graaf, "Multiparty Computations Ensuring Privacy of Each Party's Input and Correctness of the Result," *Advances in Cryptography—7th Annual International Cryptology Conference* (*CRYPTO* '87), Santa Barbara, 16-20 August 1987, pp. 87-119.

[15] D. Beaver and S. Goldwasser, "Multiparty Computation with Faulty Majority," *Proceedings of the 30th IEEE*

*Symposium on the Foundations of Computer Science*, *Research Triangle Park*, *North Carolina*, 30 October-1 November 1989, pp. 468-473.

[16] H. Y. Lin and L. Harn, "Fair Reconstruction of a Secret," *Information Processing Letters*, Vol. 55, No. 1, 1995, pp. 45-47. http://dx.doi.org/10.1016/0020-0190(95)00045-E

[17] J. Pieprzyk and X.-M. Zhang, "Cheating Prevention in Secret Sharing over $GF(p^t)$," *Progress in Cryptology—2nd International Conference on Cryptology*, Chennai, 16-20 December 2001, pp. 79-90.

[18] J. Pieprzyk and X.-M. Zhang, "On Cheating Immune Secret Sharing," *Discrete Mathematics and Theoretical Computer Science*, Vol. 6, No. 2, 2004, pp. 253-264.

[19] L. Harn, "Group-Oriented (*t*, *n*) Threshold Digital Signature Scheme and Digital Multisignature," *IEE Proceedings—Computers and Digital Techniques*, Vol. 141, No. 5, 1994, pp. 307-313. http://dx.doi.org/10.1049/ip-cdt:19941293

[20] C. Delerablee and D. Pointcheval, "Dynamic Threshold Public-Key Encryption," *Advances in Cryptography—28th Annual International Cryptology Conference* (*CRYPTO* '08), Santa Barbara, 17-21 August 2008, pp. 317-334.

[21] R. Bendlin and I. Damgard, "Threshold Decryption and Zero-Knowledge Proofs for Lattice-Based Cryptosystems," *Proceedings of 7th Theory of Cryptography Conference* (*TCC* '10), Zurich, 9-11 February 2010, pp. 201-218.

[22] L. Ertaul and W. Lu, "ECC Based Threshold Cryptography for Secure Data Forwarding and Secure Key Exchange in MANET (I)," *Proceedings of the 4th IFIP-TC6 International Conference on Networking Technologies, Services, and Protocols*; *Performance of Computer and Communication Networks*; *Mobile and Wireless Communication Systems*, Waterloo, 2-6 May 2005, pp. 102-113.

[23] Y. Desmedt, "Some Recent Research Aspects of Threshold Cryptography," *Proceedings of the 1st International Workshop* (*ISW* '97), Tatsunokuchi, 17-19 September 1997, pp. 158-173.

[24] Y. Desmedt, "Threshold Cryptosystems," *Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques* (*AUSCRYPT* '92), Gold Coast, Queensland, 13-16 December 1992, pp. 1-14.

[25] M. Abdalla, S. Miner and C. Namprempre, "Forward-secure Threshold Signature Schemes," *Topics in Cryptology—The Cryptographer's Track at RSA Conference* (*CT-RSA* '01), San Francisco, 8-12 April 2001, pp. 441-456.

[26] J. Baek and Y. Zheng, "Identity-Based Threshold Signature Scheme from the Bilinear Pairings," *Proceedings of the International Conference on Information Technology*: *Coding and Computing* (*ITCC* '04), Las Vegas, 5-7 April 2004, p. 124.

[27] T. A. ElGamal, "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms," *IEEE Transactions on Information Theory*, Vol. 31, No. 4, 1985, pp. 469-472. http://dx.doi.org/10.1109/TIT.1985.1057074