On Dealer-free Dynamic Threshold Schemes

Mehrdad Nojoumian[*]
Department of Computer Science
Southern Illinois University
Carbondale, IL 62901, USA

Douglas R. Stinson[†]
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, Ontario N2L 3G1, Canada

December 14, 2012

Abstract

In a threshold scheme, the sensitivity of the secret as well as the number of players may fluctuate due to various reasons, e.g., mutual trust may vary or the structure of the players' organization might be changed. A possible solution to this problem is to modify the threshold and/or change the secret. Moreover, a common problem with almost all secret sharing schemes is that they are "one-time", meaning that the secret and shares are known to everyone after a public secret recovery process. This problem could be resolved if the dealer shares various secrets at the beginning, but a better solution is to dynamically generate new secrets in the absence of the dealer. These issues are our main motivation to revisit dynamic threshold schemes.

Therefore, we first provide the first comprehensive study of threshold modification techniques in both the passive and active adversary models. We first review an existing method for threshold modification based on resharing shares of a secret; this method is secure in the setting of a passive adversarial coalition. We then discuss two methods, termed *public evaluation* (for threshold reduction) and *zero addition* (for threshold increase) that can be used in both the passive and active adversarial setting. In the case of an active adversary, the techniques make use of *verifiable secret sharing schemes*, whereas the schemes considered in the passive adversary model are all based on the *Shamir scheme*. As an application, we discuss how the threshold and the secret can be changed multiple times to arbitrary values after the scheme's initialization.

Keywords: secret sharing, unconditional security, threshold and secret changeability

1 Introduction

In a *secret sharing scheme*, a secret is divided into shares by a dealer for distribution among a set of players, and then an authorized subset of players collaborate to recover the secret [22, 5]. We first provide the definition of an "access structure".

Definition 1.1. *Let $\mathcal{P}$ be a finite set of* players. *An* access structure Γ *is a set of subsets of players (called* authorized subsets*) that satisfies two conditions:*

(a) if $A \in \Gamma$ and $A \subseteq B \subseteq \mathcal{P}$, then $B \in \Gamma$, and

[*]M. Nojoumian's research was supported by NSERC Canada Graduate Scholarship
[†]D. Stinson's research was supported by NSERC discovery grant 203114-11

(b) if $A \in \Gamma$ then $|A| > 0$.

In a threshold access structure*, the authorized sets are all sets of players A such that $|A| \geq t$, where t is the* threshold.

In a secret sharing scheme with access structure Γ, each player in $\mathcal{P}$ has a share. If the players in an authorized subset combine their shares, then they can compute the secret. On the other hand, if the players in an unauthorized subset combine their shares, then they have no information about the value of the secret. In particular, in a (t, n)*-threshold secret sharing scheme*, any t players can combine their shares to reveal the secret, but no set of $t - 1$ players can learn any information about the secret.

One realization of a threshold scheme is a *Shamir scheme* [22]. Let q be prime. The dealer $\mathcal{D}$ selects a random polynomial $f(x) \in \mathbb{Z}_q[x]$ of degree at most $t-1$ such that its constant term is the secret, i.e., $f(0) = \alpha$. He then sends the share $f(i)$ to player P_i, for $1 \leq i \leq n$. Subsequently, any subset Δ of at least t players can cooperate to reveal the secret by Lagrange interpolation:

$$f(0) = \sum_{i \in \Delta} \Bigg(\prod_{j \in \Delta, j \neq i} \frac{j}{j-i} \times f(i) \Bigg). \tag{1}$$

For future use, let

$$\gamma_i^{\Delta} \overset{\text{def}}{=} \prod_{j \in \Delta, j \neq i} \frac{j}{j-i}$$

denote the *Lagrange constants* used in the above formula, where $i \in \Delta$.

In a *verifiable secret sharing scheme* (or *VSS*) [7], players are able to verify the consistency of their shares after the scheme's initialization. They can also verify the correctness of their shares in other phases of the scheme's operation. There have been many constructions of VSS over the years, e.g., see [10, 11, 21, 24]. A survey of several VSS schemes is found in the introduction of [10].

The closely related notion of *proactive secret sharing scheme* (or *PSS*) is introduced in [12]. In a PSS, the shares of players are updated without changing the secret. This "share update" is done by adding some polynomials with zero constant terms to the original secret sharing polynomial. In fact, PSS is proposed to deal with a *mobile adversary* who collects the shares of an increasing number of players over time in order to recover the secret. We should stress that PSS is different from a *dynamic threshold scheme*, where the parameters of the protocol (such as the threshold, the number of players, etc.) can be changed after the scheme's initialization.

We consider various types of adversaries in our constructions. In the *passive adversary* setting, the players follow protocols correctly but unauthorized sets may attempt to learn the secret; such adversaries are also known as *honest-but-curious.* Further, in the passive adversary setting, members of a coalition might exchange information secretly with each other. On the other hand, in the *active adversary* model, the players may deviate from protocols in arbitrary ways, e.g., by transmitting incorrect information (for instance, to prevent the secret's recovery or to cause the reconstruction of an incorrect secret) while at the same time trying to learn the secret, or by broadcasting secret information, e.g., shares, to all the players.

In addition, a passive or active adversary can be *static* or *mobile.* The former refers to an adversary who corrupts a fixed set of players ahead of time, while in the latter case, the adversary may corrupt different players at various stages of the protocol's execution. Finally, the security model can be *computational*, where the security of the protocol relies on computational assumptions,

Secret Sharing

1. The dealer $\mathcal{D}$ selects a symmetric polynomial $f(x, y) \in \mathbb{Z}_q[x, y]$ of degree $t - 1$, where $f(0, 0)$ is the secret α. He sends $f_i(x) = f(x, \omega^i)$ to P_i for $1 \leq i \leq n$.

2. P_i and P_j perform *pairwise checks*, i.e., they verify that $f_i(\omega^j) = f_j(\omega^i)$. If P_i finds that $f_i(\omega^j) \neq f_j(\omega^i)$, he broadcasts the ordered pair (i, j) to *accuse* P_j.

3. Each P_i computes a subset $\Gamma \subseteq \{1, ..., n\}$ such that any ordered pair $(i, j) \in \Gamma \times \Gamma$ is not broadcasted. If $|\Gamma| \geq n - (t - 1)$, P_i outputs $ver_i = 1$; otherwise, $ver_i = 0$.

4. The sharing is accepted if at least $n - (t - 1)$ players output $ver_i = 1$; otherwise, $\mathcal{D}$ is disqualified. Note that Γ can be constructed by a polynomial time algorithm (see [10]).

Secret Recovery by a player P_j

1. Each player P_i sends the constant term of his share, that is, $f_i(0)$, to a selected player P_j.

2. Player P_j computes a polynomial $f_j(y)$ such that $f_j(\omega^i) = f_i(0)$ for at least $n-2(t-1)$ values of i. He then computes the secret $f_j(0) = f(0, 0)$.

Figure 1: An Unconditionally Secure Verifiable Secret Sharing Scheme

e.g., the hardness of factoring, or *unconditional*, when the adversary is allowed to have unlimited computational power.

1.1 Review of a Simple VSS Scheme

The authors in [24] propose a VSS scheme in which players check the consistency of their shares by symmetric polynomials. This protocol uses pairwise channels as well as a broadcast channel, and it is secure under the assumption that $|\nabla| \leq t - 1 \leq \lfloor \frac{n-1}{4} \rfloor$ where ∇ denotes the set of bad players. If $|\nabla| \leq \lfloor \frac{n-1}{4} \rfloor$, then a suitable error correction technique, e.g., based on a Reed-Solomon code, can be used to recover the secret correctly. All computations are done in $\mathbb{Z}_q$ (where q is a prime number) and ω is a primitive element in this field.

For future reference, this scheme is presented in Figure 1. We will be modifying this scheme in later sections of this paper in the presentation some of our new protocols. We note that these protocols could be described in terms of other VSS schemes, as well.

The scheme satisfies the following properties (for proofs, see [24]):

1. If a good player P_i outputs $ver_i = 0$ at the end of the "sharing" phase, then every good player outputs $ver_i = 0$. If this occurs, then more than $t - 1$ shares have been corrupted by malicious players and a dishonest dealer.

2. If the dealer is honest, then $ver_i = 1$ for every good P_i. In this situation, at most $t - 1$ shares might be corrupted by malicious players.

3. If at least $n-(t-1)$ players P_i output $ver_i = 1$ at the end of the sharing phase, then $\alpha' \in \mathbb{Z}_q$ will be reconstructed in the "recovery" phase, and $\alpha' = \alpha$ if the dealer is honest.

4. If $|\mathbb{Z}| = q$, α is chosen randomly from $\mathbb{Z}$, and the dealer is honest, then no coalition of at most $t-1$ players can guess the value α with probability greater than $\frac{1}{q}$ at the end of the sharing phase.

1.2 Threshold Changeability

The goal of *threshold changeability* is to convert a (t,n)-threshold scheme into a (t',n)-threshold scheme, where we allow both $t < t'$ (*threshold increase*) and $t' < t$ (*threshold reduction*). We should note that, in the case of threshold increase, if the decision is to keep the same secret, it has been observed (e.g., see [16]) that at least $n-t+1$ players have to erase their old shares honestly (otherwise, the secret can be constructed by an unauthorized set from "old" shares). Therefore, in this paper, we assume all the players always erase their old shares when updating shares.

We now recall the Lagrange interpolation method for polynomials. Let q be a prime number. Let $x_1, ..., x_t$ and $f_1, ..., f_t$ be distinct elements in the field $\mathbb{Z}_q$. Then there is a unique $f(x) \in \mathbb{Z}_q[x]$ of degree at most $t-1$ such that $f(x_i) = f_i$ for $1 \le i \le t$:

$$f(x) = \sum_{i=1}^{t} \Bigg(\prod_{1 \le j \le t, j \ne i} \frac{x - x_j}{x_i - x_j} \times f_i \Bigg). \tag{2}$$

Lagrange interpolation can also be applied to bivariate polynomials. Let $y_1, ..., y_t$ be distinct elements in $\mathbb{Z}_q$ and let $f_1(x), ..., f_t(x)$ be polynomials of degree at most $t-1$ in $\mathbb{Z}_q[x]$. Then there is a unique polynomial $f(x,y) \in \mathbb{Z}_q[x,y]$ of degree at most $t-1$ such that $f(x,y_i) = f_i(x)$ for $1 \le i \le t$:

$$f(x,y) = \sum_{i=1}^{t} \Bigg(\prod_{1 \le j \le t, j \ne i} \frac{y - y_j}{y_i - y_j} \times f_i(x) \Bigg). \tag{3}$$

When we say that a bivariate polynomial $f(x,y)$ has *degree at most* $t-1$, we mean that any term of the form cx^iy^j in $f(x,y)$ has $i \le t-1$ and $j \le t-1$.

1.3 Our Motivation and Contributions

We are motivated to revisit dynamic threshold schemes due to various issues. As stated by Martin *et al.* [17], in a threshold scheme, the sensitivity of the secret as well as the number of players may fluctuate due to various reasons. For instance,

(a) mutual trust might be decreased over time, perhaps because of organizational problems or security incidents, and

(b) the structure of the organization to which the players belong might be changed, for example, new players may join the organization and need to be incorporated into the security structure, or current parties may leave the organization while still retaining a copy of their shares.

In both of these cases, the threshold should be adjusted accordingly. In other words, modifying the threshold and/or changing the secret might be required throughout the lifetime of a secret.

This requirement is more significant when the lifetime is increased. In the literature, there exist some techniques to address this problem, but most existing solutions suffer from one or more of the following drawbacks:

- They assume the existence of an online trusted authority (i.e., an online dealer).
- They have a large storage requirement (e.g., each user needs to store multiple shares).
- They are limited to predefined modifications.

Another well-known problem with many secret sharing schemes is that they are one-time, meaning that, after recovering the secret in a public computation, the secret and the shares are known to everyone (of course, if the reconstruction process is not public, then a secret can be re-used). To resolve this problem, it would be useful to generate new secrets in the absence of an online dealer.

We review previous literature on dynamic threshold schemes in Section 1.4. In later sections, we provide various techniques for dynamic secret sharing that are more general and/or simpler than existing solutions. Indeed, our proposed methods have various desirable properties:

- Our schemes are *dealer-free*, which means the players can change the secret and the threshold after the scheme's initialization without the participation of an online dealer. (Note that this is a different concept than *mutually trusted authority-free*, which denotes a situation where there is no dealer at all, even during initialization. This idea was first discussed in [13]; see [14] for an extensive treatment of these types of schemes.)
- Our schemes are *unconditionally secure* in that they do not rely on any computational assumptions.
- Our schemes have *minimum storage cost* since players do not need to store extra shares beforehand in order to modify the parameters of the scheme at a later time.
- Finally, our schemes are *flexible* because the secret and the threshold can be changed to arbitrary values multiple times.

We should note that several of the protocols discussed in this paper make use of known techniques, but we often employ new tweaks to adapt the techniques to the new settings we consider.

In Section 2, we consider the passive adversary setting. First, we briefly discuss a commonly-used "resharing technique", also known as 2-level sharing, that modifies the threshold to any arbitrary value. This approach applies the Lagrange method in order to form a certain linear combination of players' shares after resharing (alternatively, this linear combination can (equivalently) be carried out by using a Vandermonde matrix). Next, we extend a folklore method to reduce the threshold based on revealing one or more shares. In the new variation we propose, the players first construct and then reveal an "extra" share, and then they use this extra share to modify their existing shares so that the threshold is decreased but the secret remains unchanged. We call this new technique *public evaluation*. The last technique discussed in this section is termed *zero addition*, where the players jointly construct shares of a random polynomial with constant term equal to 0, and add these shares to the shares they possessed previously. This allows the threshold to be increased.

In Section 3, we turn to the active adversary model, where the problem is more difficult and where there is little in the way of previous research. It has been observed that "obvious" modifications of resharing techniques are insecure in the active adversarial model (this is discussed in [19]).

However, we show that the techniques of public evaluation and zero addition can be generalized in a secure manner to the active adversary setting by using verifiable secret sharing schemes. We also provide methods to change both the secret and the threshold simultaneously.

Finally, Section 4 presents a short summary of the work in this paper.

1.4 Previous Work on Dynamic Threshold Schemes

There have been several approaches used in the literature to construct dynamic threshold schemes. For a good survey on this topic, see Martin [16]. In this section, we summarize some of the more popular techniques.

Several papers have made use of the method of *resharing*, wherein a subset of players each act as a dealer to reshare their share of the secret among other players in the scheme. This approach was proposed in [3] for "degree reduction", which is used in secure multiparty computation protocols to adjust the threshold after shares of two secrets are multiplied together to obtain shares of the product of the two secrets. Other papers that discuss resharing methods include [2, 3, 9, 11, 13, 18]. The most general treatment of resharing is found in the unpublished paper by Desmedt and Jajodia [9].

One important issue in resharing is to devise a technique so that players do not have to store multiple shares after resharing has taken place. This is typically accomplished by forming a suitable linear combination of these shares so they can be replaced by a single share. This linear combination can be computed as an application of the Lagrange formula, or alternatively, by a multiplication by the inverse of a Vandermonde matrix.

Various papers have discussed the use of *prepositioned schemes*, where a sequence of thresholds and/or access structures are determined when the scheme is initialized. Examples of this approach include [15, 17]. Other schemes have enabled a dynamic update of the access structure by means of a *broadcast* made by the dealer, e.g., [1, 6, 17]. Finally, there are some papers, such as [23, 25], that make use of *random noise* and/or *fake shares* to permit the threshold to be increased.

2 Schemes in the Passive Adversary Model

First, we quickly review a resharing technique that modifies the threshold to any arbitrary value using a share combining technique based on Lagrange interpolation (for a more general approach that applies to general linear secret sharing schemes, see [9]). We then observe that share combining can equivalently be accomplished by using a Vandermonde matrix, as was done, for example, in [3, 11].

Next, we present our new method of public share evaluation, where we modify and extend a folklore technique to reduce the threshold based on revealing a share (this folklore method has been suggested, for example, for disenrollment in a threshold scheme [4]). In our new variation on this theme, the players first construct and then reveal an "extra" share, and then they use this extra share to modify their existing shares so that the threshold is decreased but the secret remains unchanged.

Then, we discuss the method of *zero addition*, where the players jointly construct shares of a random polynomial with constant term equal to 0, and add these shares to the shares they possessed previously. This allows the threshold to be increased.

Finally, we compare the schemes we have described in terms of their communication complexity.

2.1 Threshold Modification by the Lagrange Method

Suppose a dealer initiates a Shamir secret sharing scheme and then leaves. That is, he randomly generates $f(x) \in \mathbb{Z}_q[x]$ of degree at most $t-1$ in which its constant term is the secret $f(0) = \alpha$, and then sends share $f(i)$ to player P_i for $1 \leq i \leq n$. Then each player reshares his share using a new random polynomial of degree at most $t'-1$. The detailed description of the scheme is given in Figure 2.

Theorem 2.1. *The threshold modification protocol presented in Figure 2 is secure under the passive adversary model, and it correctly computes the secret α.*

Proof. It is clear that a set ∇ of colluders, where $|\nabla| \leq t-1$, is not able to recover the secret after the initial distribution of shares by the dealer. In the next stage, the players reshare their shares using polynomials $g_i(x)$ of degree $t'-1$. As a result, $t'-1$ (or fewer) colluders cannot reconstruct any of these resharing polynomials in order to reveal the shares of the good players. Since all players erase their old shares after computing the new ones, there is no way to construct the secret by making use of the original shares. Therefore, the protocol is secure under the passive (honest-but-curious) adversary model.

Now, we show the correctness of the scheme, where $|\Delta| \geq t$ and $|\Delta'| \geq t'$. The following proof is similar to the proof in Section 4 of the unpublished paper [9]; the proof is included here for completeness:

$$\begin{aligned}
\sum_{j\in\Delta'} \left(\gamma_j^{\Delta'} \times \varphi_j\right) &= \sum_{j\in\Delta'} \left(\gamma_j^{\Delta'} \times \sum_{i\in\Delta} \left(\gamma_i^{\Delta} \times g_i(j)\right)\right) \qquad \text{by (5)} \\
&= \sum_{i\in\Delta} \left(\gamma_i^{\Delta} \times \sum_{j\in\Delta'} \left(\gamma_j^{\Delta'} \times g_i(j)\right)\right) \\
&= \sum_{i\in\Delta} \left(\gamma_i^{\Delta} \times g_i(0)\right) \qquad \text{by (1)} \\
&= \sum_{i\in\Delta} \left(\gamma_i^{\Delta} \times f(i)\right) \\
&= f(0) \qquad \text{by (1)} \\
&= \alpha.
\end{aligned}$$

□

2.2 Threshold Modification by a Vandermonde Matrix

In this section, we present an alternative method for threshold modification where the secret remains the same. The idea is to reshare the existing shares of the players to change the threshold from t to t', and then use (the inverse of) a Vandermonde matrix to combine the shares. We present the protocol in Figure 3.

We are not going to discuss this protocol in detail. Mainly we just want to point out that this approach is basically equivalent to the protocol from the previous subsection that uses the Lagrange method. (This equivalence is rather obvious, but it does not seem to be explicitly stated in the

Threshold Modification

1. A set Δ is determined such that it consists of the identifiers of at least t elected players. Each player $P_i \in \Delta$ selects a random polynomial $g_i(x)$ of degree at most $t' - 1$ such that $g_i(0) = f(i)$. He then gives $g_i(j)$ to P_j for $1 \leq j \leq n$, i.e., resharing the original shares by auxiliary shares.

2. The following public constants are computed:

$$\gamma_i^{\Delta} = \prod_{j \in \Delta, j \neq i} \frac{j}{j-i} \quad \text{for all } i \in \Delta. \tag{4}$$

3. Each player P_j $(1 \leq j \leq n)$ erases his old shares, and then combines the auxiliary shares he has received from other players to compute his new share as follows:

$$\varphi_j = \sum_{i \in \Delta} \left(\gamma_i^{\Delta} \times g_i(j) \right). \tag{5}$$

Secret Recovery

- Now, if a set Δ' of at least t' players P_j cooperate, they can recover α by using the Lagrange interpolation method:

$$\alpha = \sum_{j \in \Delta'} \left(\gamma_j^{\Delta'} \times \varphi_j \right). \tag{6}$$

Figure 2: Threshold Modification by the Lagrange Method in the Passive Adversary Model

existing literature.) The only difference is that the Lagrange method uses the shares belonging to any set of at least t players to compute the updated shares in the scheme, while the Vandermonde method uses the shares of all n players. This is less efficient, of course.

2.3 Threshold Decrease by Public Evaluation

As we mentioned previously, a folklore method to reduce the threshold in a secret sharing scheme consists of revealing one or more shares However, there are some issues that make this approach inconvenient:

- If a player reveals his share, then he effectively removes himself as a player in the secret sharing scheme.
- A dealer can easily construct and reveal a "new" share to the players, but this requires an online dealer.
- In any event, the remaining players have to store the revealed share as well as their old share (however, we note that the revealed share does not need to be kept secret).

Threshold Modification

1. The resharing phase is similar to the first step of the previous protocol, which was presented in Figure 2.

2. Participants then compute the first row of a public matrix $\mathcal{V}_{n\times n}^{-1} \pmod{q}$ to adjust the threshold, where $\mathcal{V}_{n\times n}$ is the Vandermonde matrix, i.e., $\mathcal{V}_{i,j} = i^{(j-1)}$ for $1 \leq i, j \leq n$. Suppose this vector is $\mathcal{V}_{1\times n}^{-1} = (v_1, \quad v_2, \quad \ldots, \quad v_n)$.

3. Eventually, each player P_j computes his final share by multiplying $\mathcal{V}_{1\times n}^{-1}$ by his vector of shares:
$$\varphi(j) = \sum_{i=1}^{n} v_i \, g_i(j).$$

Secret Recovery

- To recover the secret, t' participants P_j have to collaborate in order to construct a polynomial of degree $t' - 1$:
$$\varphi(x) = \sum_{j=1}^{t'} \Bigg(\prod_{1\leq i \leq t', i\neq j} \frac{x-i}{j-i} \times \varphi(j) \Bigg).$$
Then they compute the secret $\varphi(0)$. (Alternatively, the secret can be reconstructed directly using the Lagrange Interpolation Formula, without first computing $\varphi(x)$.)

Figure 3: Threshold Modification by a Vandermonde Matrix in the Passive Adversary Model

Our new approach enables the players to first jointly construct and then reveal an "extra" share, and then use this extra share to modify their existing shares so that the threshold is decreased by one, but the secret remains unchanged. The players can use the *enrollment protocol* of [20] to publicly generate a new share at a new point different from the players' existing *ids*. Each player then combines the revealed share with his own private share in order to decrease the threshold. For instance, suppose P_1, P_2, P_3 have three shares at points $x = 1, 2, 3$ (respectively) on a polynomial f of degree 2. If they jointly construct and reveal a share at a new point, say $x = 4$, they can then combine this share with their own private shares such that the degree of f is decreased by 1 but the secret remains unchanged; we shortly explain how to perform this combination.

Suppose $f(x) \in \mathbb{Z}_q[x]$ of degree $t-1$ is the secret sharing polynomial. Let $\Gamma \subseteq \mathbb{Z}_q \backslash \{0\}$ denote the set of players' *ids*. As a result, each P_i for $i \in \Gamma$ receives the share $f(i) \in \mathbb{Z}_q$ from the dealer. For the sake of simplicity, suppose the players want to decrease the threshold from t to $t-1$ in the absence of the dealer (for further threshold reduction, they can just repeat the same procedure again). The proposed protocol is presented in Figure 4.

Theorem 2.2. *The threshold reduction approach presented in Figure 4 is secure under the passive adversary model, and it correctly reduces the threshold.*

Proof. The security of the first four steps is shown in [20]. We just need to show that the new shares are on a polynomial $\hat{f}(x)$ of degree at most $t-2$ where $f(0) = \hat{f}(0)$. Assuming that the new

Threshold Decrease

1. The players select an *id* j such that $j \notin \mathcal{P}$. Subsequently, t players P_i are selected (e.g., $1 \leq i \leq t$). They compute Lagrange constants as follows:
$$\gamma_i = \prod_{1 \leq k \leq t, i \neq k} \frac{j-k}{i-k}.$$
2. Each P_i multiplies his share $f(i)$ by his Lagrange constant. He then randomly splits the result into t portions, i.e., $f(i) \times \gamma_i = \partial_{1i} + \partial_{2i} + \cdots + \partial_{ti}$ for $1 \leq i \leq t$.
3. The players exchange ∂_{ki}-s through pairwise channels (in a fashion similar to Figure 2). As a result, each P_k holds t values. He adds them together and reveals $\sigma_k = \sum_{i=1}^{t} \partial_{ki}$ to everyone.
4. The players add these values σ_k for $1 \leq k \leq t$ together to compute the public share $f(j) = \sum_{k=1}^{t} \sigma_k$.
5. Each P_i combines his private share $f(i)$ with the public share $f(j)$ as follows:
$$\hat{f}(i) = f(j) - j\left(\frac{f(i) - f(j)}{i - j}\right). \tag{7}$$
6. The shares $\hat{f}(i)$ are on a new polynomial $\hat{f}(x) \in \mathbb{Z}_q[x]$ of degree $t-2$ where $\hat{f}(0) = f(0)$. Therefore, $t-1$ players are now sufficient to recover the secret.

Figure 4: Threshold Decrease by Public Evaluation in the Passive Adversary Model

share $(j, f(j))$ has been revealed, we have:
$$(x - j) \mid \big(f(x) - f(j)\big),$$
where "$|$" denotes division of polynomials. Let us define $f^*(x)$ of degree $t-2$ as follows:
$$f^*(x) \overset{\text{def}}{=} \frac{f(x) - f(j)}{x - j}. \tag{8}$$
Using (8), the share of each player P_i on f^* is obtained as follows:
$$f^*(i) = \frac{f(i) - f(j)}{i - j}. \tag{9}$$
Accordingly, using (8), the secret associated with f^* will be:
$$f^*(0) = \frac{f(0) - f(j)}{-j}. \tag{10}$$

Polynomial Production

1. Initially, t players P_i are selected at random in order to act as independent dealers; they each might be honest or malicious.

2. Each of the t chosen players P_i shares a secret, say δ_i, among all the players using a Shamir scheme, where the degree of the secret sharing polynomial is $t-1$. Then all players have shares of every secret δ_i.

3. Every player adds his shares of the δ_i-s together. As a result, each player has a share on a polynomial $g(x)$ of degree $t-1$ with a constant term $\delta = \sum \delta_i$.

Figure 5: Generating Shares of a Random Number δ in the Passive Adversary Model

Using (10), the relation between the old secret $f(0)$ and the new secret $f^*(0)$ is obtained:

$$f(0) = f(j) - jf^*(0). \tag{11}$$

Suppose we define the new polynomial

$$\hat{f}(x) = f(j) - jf^*(x) = f(j) - j\left(\frac{f(x) - f(j)}{x - j}\right). \tag{12}$$

Then $\hat{f}(x)$ has degree at most $t-2$ and $\hat{f}(0) = f(0)$. It is easy for each player P_i to compute their new share $\hat{f}(i)$:

$$\hat{f}(i) = f(j) - j\left(\frac{f(i) - f(j)}{i - j}\right). \tag{13}$$

Therefore, each P_i privately computes $\hat{f}(i)$ to update his share. As a result, the threshold is decreased while the secret remains the same. □

2.4 Threshold Increase by Zero Addition

The idea of the next protocol we present is to increase the threshold by constructing shares of a polynomial that corresponds to a secret having the value zero and threshold $t' > t$, and adding these new shares to the players' current shares. We call this threshold increase by *zero addition*. We note that this idea is also employed in proactive secret sharing [12], except that the threshold is not changed in that setting. There are two steps involved in the zero addition protocol. The first step consists of a straightforward process, which we call *polynomial production*, that enables the players to collectively generate shares of an unknown secret δ without a dealer (this idea was first discussed in [13] and is treated in a more general setting in [14]). The second step adjusts the polynomial that is used to share the unknown secret value δ so that the participants instead have shares of 0. In this protocol, we assume that at most $t-1$ players are malicious. The protocol is presented in Figure 5.

In the first step of polynomial production, each of t participants P_i acts as an independent dealer and distributes shares of a secret δ_i using a Shamir scheme. Since we have t dealers, δ

Threshold Increase

For every player P_i, suppose $f(i)$ is the share of an unknown secret α belonging to P_i.

1. Players use polynomial production to generate shares of an unknown secret δ on a polynomial $g(x)$ of degree $t' - 2$.

2. Each player P_i multiplies his share $g(i)$ by i. Now, each P_i has a share of 0 on the polynomial $\hat{g}(x) = xg(x)$ of degree $t' - 1$.

3. Each player adds his share $f(i)$ of α to his share $ig(i)$ of 0. As a result, each player has a share of α, where the new threshold is $t' > t$.

Figure 6: Increasing the Threshold by Zero Addition in the Passive Adversary Model

remains secret even if $t - 1$ players collaborate and combine their secret shares. Therefore, the value δ constructed in step 3 remains unknown to everyone.

Now, suppose each player initially has a share of the secret α on a polynomial of degree $t - 1$. We will show how to use the polynomial production protocol in order to increase the threshold *without changing the secret* α. Our zero addition protocol for threshold increase is presented in Figure 6. The trick used in this protocol is to adjust the polynomial $g(x)$ that is used to share the unknown secret value δ so that the participants instead have shares of 0. This can be done using a simple technique similar to the one used in [8, p. 357]. If we define $\hat{g}(x) = xg(x)$, then $\hat{g}(0) = 0$. Furthermore, each player can adjust his share in an appropriate way: $\hat{g}(i) = ig(i)$.

After executing the above protocol, the secret remains the same while the threshold is increased to t'.

2.5 Comparison

In this section, we briefly compare the various methods for threshold change that we have discussed in this section. Since the computations to be performed are relatively straightforward, we concentrate on the *communication complexity*, i.e., the number of messages that need to be exchanged by pairs of players in the scheme. First, we consider the Lagrange based resharing method given in Figure 2. Here, t players each construct n shares and distribute them, so the communication complexity is $\Theta(tn)$. The zero addition protocol from Figure 6 also involves t players distributing shares among the n players, so the communication complexity is again $\Theta(tn)$. The public evaluation scheme from Figure 4 is somewhat different. The first part of this protocol consists of the construction of a new share via the enrollment protocol from [20]. This protocol has a communication complexity of $\Theta(t^2)$. Then the new share has to be communicated to all the players in the scheme, so the overall communication complexity is $\Theta(t^2 + n)$. This process just reduces the threshold by 1; to reduce the threshold from t to t' (where $t' < t$), the protocol has to be repeated $t - t'$ times, and the overall complexity is $\Theta((t - t')(n + t^2))$. This complexity is still $O(tn + t^3)$, however. Note that this method might be the most efficient if t is small compared to n and t' is close to t.

Threshold Decrease

1. The players randomly select a new *id* j such that $j \notin \mathcal{P}$. Each player P_i then computes and reveals $f_i(\omega^j)$ to all the other players.

2. The players compute the new share $f_j(x)$ by using points $(i, f_i(\omega^j))$ (see [24, Section 4.3]).

3. Each P_i combines his private share $f_i(x)$ with the new share $f_j(x)$ as follows:

$$\hat{f}_i(x) = \hat{f}(x, \omega^i) = (\omega^j)^2 \left(\frac{f_i(x) - f_j(x) - f_j(\omega^i) + f_j(\omega^j)}{(x - \omega^j)(\omega^i - \omega^j)} \right) + 2f_j(0) - f_j(\omega^j). \quad (14)$$

4. Shares $\hat{f}_i(x) \in \mathbb{Z}_q[x]$ correspond to a symmetric bivariate polynomial $\hat{f}(x, y) \in \mathbb{Z}_q[x, y]$ of degree $t - 2$ that has the same secret as before. Therefore, $t - 1$ players are now sufficient to recover the secret.

Figure 7: Threshold Decrease by Public Evaluation in the Active Adversary Model

3 Schemes in the Active Adversary Model

In this section, we study threshold modification in the active adversary setting. We note that there are problems with adapting resharing methods to handle active active adversaries, e.g., see [19] for a good discussion of these issues. Therefore, we provide two protocols not based on resharing, for threshold increase and decrease respectively, which are generalizations of the protocols for the passive adversary model that we presented in Sections 2.3 and 2.4. In addition to the private channels between each pair of players, a synchronous broadcast channel is assumed to exist. All computations are again performed in a finite field $\mathbb{Z}_q$, where q is a prime number.

3.1 Threshold Decrease by Public Evaluation

The original secret sharing polynomial $f(x, y) \in \mathbb{Z}_q[x, y]$ of degree at most $t - 1$ is a symmetric bivariate polynomial. Each player P_i receives $f_i(x) = f(x, \omega^i) \in \mathbb{Z}_q[x]$ from the dealer. For the sake of simplicity, suppose the players want to decrease the threshold from t to $t - 1$ in the active adversary model and in the absence of the dealer (for further threshold reduction, they can repeat the procedure). The players can first use the *recovery* protocol proposed in [24, Section 4.3] to publicly generate an extra share $f_j(x)$ at a desired point ω^j where j is different from the existing players' *ids*. This is similar to what we did in Figure 4 in the passive adversary case. Next, the players need to combine the revealed share with their own private shares; this is more complicated now that we are in the active adversary model, due to the use of symmetric bivariate polynomials. The protocol to accomplish this is presented in Figure 7.

Theorem 3.1. *The threshold reduction protocol presented in Figure 7 is secure under the active adversary model, and it correctly reduces the threshold.*

Proof. If at most $t - 1$ malicious parties reveal incorrect values in the first step, the players can correctly compute $f_j(x)$ in the second step through an error correction technique (see Section 1.1 for details). It is also easy to show that all the bivariate polynomials remain symmetric during the third and fourth steps. As a result, the players can perform pairwise checks in order to detect any malicious behavior. We just need to show that the new shares are on a polynomial $\hat{f}(x,y)$ of degree at most $t-2$, where $\hat{f}(0,0) = f(0,0)$. We know that

$$(y-\omega^j) \mid \big(f(x,y) - f_j(x)\big)$$

and

$$(y-\omega^j) \mid \big(f_j(y) - f_j(\omega^j)\big).$$

Now, it is clear that, if $a \mid b$ and $a \mid c$, then $a \mid (b-c)$. As a result, we obtain:

$$(y-\omega^j) \mid \big(f(x,y) - f_j(x) - f_j(y) + f_j(\omega^j)\big).$$

Since the polynomial $f(x,y) - f_j(x) - f_j(y) + f_j(\omega^j)$ is symmetric in x and y, $(x-\omega^j)$ also divides it. Now, if $a \mid c$, $b \mid c$, and $gcd(a,b) = 1$, then $ab \mid c$. Clearly, $(x-\omega^j)$ and $(y-\omega^j)$ do not have a common divisor. Therefore, we obtain

$$(x-\omega^j)(y-\omega^j) \mid \big(f(x,y) - f_j(x) - f_j(y) + f_j(\omega^j)\big).$$

We define a symmetric polynomial $f^*(x,y)$ of degree $t-2$ as follows:

$$f^*(x,y) \stackrel{\text{def}}{=} \frac{f(x,y) - f_j(x) - f_j(y) + f_j(\omega^j)}{(x-\omega^j)(y-\omega^j)}. \tag{15}$$

Accordingly, using (15), the secret associated with f^* will be:

$$f^*(0,0) = \frac{f(0,0) - f_j(0) - f_j(0) + f_j(\omega^j)}{(\omega^j)^2}. \tag{16}$$

Using (16), the relation between the old secret and the new one is obtained:

$$f(0,0) = (\omega^j)^2 f^*(0,0) + 2f_j(0) - f_j(\omega^j). \tag{17}$$

Now, suppose we define another symmetric polynomial $\hat{f}(x,y)$, of degree at most $t-2$, as follows:

$$\begin{aligned} \hat{f}(x,y) &= (\omega^j)^2 f^*(x,y) + 2f_j(0) - f_j(\omega^j) && (18)\\ &= (\omega^j)^2 \left(\frac{f(x,y) - f_j(x) - f_j(y) + f_j(\omega^j)}{(x-\omega^j)(y-\omega^j)}\right) + 2f_j(0) - f_j(\omega^j). && (19) \end{aligned}$$

From (19), we see that each P_i can privately compute $\hat{f}_i(x) = \hat{f}(x,\omega^i)$ to update his share:

$$\hat{f}(x,\omega^i) = (\omega^j)^2 \left(\frac{f(x,\omega^i) - f_j(x) - f_j(\omega^i) + f_j(\omega^j)}{(x-\omega^j)(\omega^i-\omega^j)}\right) + 2f_j(0) - f_j(\omega^j).$$

Furthermore, from (18) and (17), it is clear that $f(0,0) = \hat{f}(0,0)$, so the secret remains the same. □

Polynomial Production

1. Initially, $t+1$ players P_i are selected at random in order to act as independent dealers; they each might be honest or malicious.

2. Each P_i shares a secret, say δ_i, among all the players using a VSS scheme where the degree of the secret sharing polynomial is $t-1$. If the sharing is accepted, then all good players have consistent shares of the secret δ_i.

3. Each P_i adds shares of the accepted δ_i-s together. As a result, each player has a share on a symmetric polynomial $g(x, y)$ of degree $t-1$ with a constant term $\delta = \sum \delta_i$.

Figure 8: Generating Shares of a Random Number δ in the Active Adversary Model

3.2 Threshold Increase by Zero Addition

We now discuss how to modify the idea of "zero addition" from Section 2.4 to apply to the active adversary setting, assuming that at most $t-1$ players are actively malicious. There are two main changes to the approach used in the passive adversary setting that was discussed in Section 2.4. First, we need to replace basic Shamir schemes by VSS schemes, in order to detect actively malicious adversaries. The second change is to use $t+1$ (rather than t) dealers during polynomial production. The resulting protocol for polynomial production is presented in Figure 8.

In the first step of polynomial production, each of $t+1$ participants P_i acts as an independent dealer and distributes shares of a secret δ_i using a VSS. Each of these P_i's may corrupt at most $t-1$ shares; otherwise, he will be disqualified. As a result, the polynomial production protocol works under the assumptions of the VSS that is being used. If a player is disqualified in the second step, the other players simply discard the shares that he has distributed. Since we have $t+1$ dealers, δ remains secret even if $t-1$ of them reveal their δ_i's. In this situation, since $(t+1)-(t-1)=2$, there remain two δ_i's that are not publicly revealed. Therefore, the value δ constructed in step 3 remains unknown to everyone.

Now, in the zero addition protocol, we adjust the (symmetric) polynomial $g(x, y)$ that is used to share the unknown secret value δ so that the participants instead have shares of 0. If we define $\hat{g}(x, y) = (x+y)g(x, y)$, then $\hat{g}(x, y)$ is still symmetric and $\hat{g}(0, 0) = 0$. Furthermore, each player can adjust his share in an appropriate way: $\hat{g}(x, \omega^i) = (x+\omega^i)g(x, \omega^i)$. The resulting zero addition protocol is presented in Figure 9.

After executing the above protocol, the secret remains the same while the threshold is increased to t'. Since the polynomials remain symmetric during every step of the proposed protocol, the players can detect any malicious behaviour by pairwise checks through secure channels, as was done in the VSS presented in Figure 1.

3.3 Application: Changing Both the Threshold and the Secret

As an example showing the applications of the techniques we have developed, we discuss how to increase the threshold t to a new threshold $t' > t$ and simultaneously update the secret α to a new value α' that depends on α in some specified way, i.e., $\alpha' = \alpha\beta + \delta$ where $\beta, \delta \in \mathbb{Z}_q$ are unknown constants and q is prime. These can be considered as generalizations of techniques used in secure

Threshold Increase

Suppose $f(x, \omega^i)$ is the share of α belonging to P_i (see Figure 1).

1. Players use the VSS of [24] to generate shares of an unknown secret δ on a symmetric polynomial $g(x, y)$ of degree $t' - 2$ using polynomial production.

2. Each P_i multiplies his share $g(x, \omega^i)$ by $(x + \omega^i)$. Now, each P_i has a share of 0 on the symmetric polynomial $\hat{g}(x, y) = (x + y)g(x, y)$ of degree $t' - 1$.

3. Each player adds his share $f(x, \omega^i)$ of α to his share $(x + \omega^i)g(x, \omega^i)$ of 0. As a result, each player has a share of α where the new threshold is $t' > t$.

Figure 9: Increasing the Threshold by Zero Addition in the Active Adversary Model

multiparty computation, where two important primitives are to construct shares of the sum or the product of two secrets.

The updating of the secret will take place in the absence of the dealer who initially created the scheme. The methods we describe are based on verifiable secret sharing schemes and thus they are secure in the active adversary setting. To motivate our ideas, let's first adapt some well-known "folklore" methods (e.g., similar to those used in [3]) to combine shares of two existing secrets to form shares of a new secret. However, note that we are using these techniques in conjunction with a threshold increase, which is not considered in most previous works. Here, we suppose that an unknown secret α has been shared among n players using a Shamir scheme with threshold t. We have the following operations, where we assume $t' > t$:

- Let δ be an unknown secret that has been shared among the same n players, again using a Shamir scheme with threshold t'. If every player computes the sum of their two shares, then the result is a sharing of $\alpha' = \alpha + \delta$. Here, the threshold value is increased to t'.

- Let β be another unknown secret that has been shared among the same n players, again using a Shamir scheme with threshold $t' - t + 1$. If every player computes the product of their two shares, then the result is a sharing of $\alpha' = \alpha\beta$. Here, the threshold value is also increased to t'. However, it should be noted that the resulting secret sharing polynomial of degree $t' - 1$ is not a "random" polynomial (this problem was first noted in [3] in connection with the problem of secret multiplication) because it can be factored into two smaller polynomials of degrees $t - 1$ and $t' - t$, a property that will not hold for "random" polynomials. Therefore, we should not use this product construction in isolation; we should always follow it by an addition operation that increases or maintains the threshold, to guarantee that the final secret sharing polynomial is random.

The above methods of forming shares of the sum or products of two secrets assume that the players already have shares of two secrets. Here we consider a slightly different setting, where the players have shares of one secret, α, and they want to obtain shares of a new secret, $\alpha' = \alpha\beta + \delta$. The values of β and δ are unknown random numbers that are determined interactively by the set of players. Earlier, we showed how the players can collectively generate shares of an unknown secret

δ without a dealer, as shown in Figure 8. Once the players have shares of β and δ, they can first compute shares of $\alpha\beta$ using the product protocol described above. They can then compute shares of $\alpha\beta + \delta$ using the sum protocol described above. Here is a more detailed description of a process that will change the secret α to $\alpha' = \alpha\beta + \delta$, while simultaneously increasing the threshold from t to t'.

1. Use polynomial production to create shares of an unknown secret β with threshold $t' - t + 1$.
2. Each player multiplies his share of α with his share of β. As a result, the new threshold is t'.
3. Use polynomial production to create shares of an unknown secret δ with threshold t'.
4. Each player adds his share of $\alpha\beta$ and his share of δ. Now each player has a share of α' with threshold t'.

4 Conclusion

In this paper, various *dynamic threshold schemes* were discussed. We illustrated how to increase or decrease the threshold in the passive and active adversary models when the dealer is not involved in the scheme after the scheme's initialization. In addition, we showed how to change the secret and increase the threshold at the same time. Table 1 summarizes secure threshold modification techniques in the passive and active adversary models.

Threshold Change	Passive Adversary	Active Adversary
Decrease	Figure 2: resharing by Lagrange method Figure 3: resharing by Vandermonde matrix Figure 4: Public evaluation	Figure 7: Public evaluation
Increase	Figure 2: resharing by Lagrange method Figure 3: resharing by Vandermonde matrix Figure 6: Zero addition	Figure 9: Zero addition

Table 1: Summary of Secure Threshold Modification Techniques

Acknowledgments

We would like to thank the referees for their helpful and constructive comments.

References

[1] Susan G. Barwick, Wen Ai Jackson, and Keith M. Martin. Updating the parameters of a threshold scheme by minimal broadcast. *IEEE Transactions on Information Theory*, 51(2):620–633, 2005.

[2] Donald Beaver. Multiparty protocols tolerating half faulty processors. In *9th Annual International Cryptology Conference, CRYPTO*, volume 435 of *LNCS*, pages 560–572. Springer, 1989.

[3] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *20th Annual ACM Symposium on Theory of Computing, STOC*, pages 1–10, 1988.

[4] Bob Blakley, G. R. Blakley, Agnes Hui Chan, and James L. Massey. Threshold schemes with disenrollment. In *CRYPTO*, pages 540–548, 1992.

[5] G. R. Blakley. Safeguarding cryptographic keys. In *National Computer Conference*, volume 48 of *AFIPS Conference proceedings*, pages 313–317. AFIPS Press, 1979.

[6] Carlo Blundo, Antonella Cresti, Alfredo De Santis, and Ugo Vaccaro. Fully dynamic secret sharing schemes. *Theoretical Computer Science*, 165(2):407–440, 1996.

[7] Benny Chor, Shafi Goldwasser, Silvio Micali, and Baruch Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults. In *26th Annual IEEE Symposium on Foundations of Computer Science, FOCS*, pages 383–395, 1985.

[8] Paolo D'Arco and Douglas R. Stinson. On unconditionally secure robust distributed key distribution centers. In *8th Int. Conf. on the Theory and Application of Cryptology and Info. Security, ASIACRYPT*, LNCS, pages 346–363. Springer, 2002.

[9] Yvo Desmedt and Sushil Jajodia. Redistributing secret shares to new access structures and its applications. In *Technical Report ISSE TR-97-01*. George Mason U, 1997.

[10] Rosario Gennaro, Yuval Ishai, Eyal Kushilevitz, and Tal Rabin. The round complexity of verifiable secret sharing and secure multicast. In *33th Annual ACM Symposium on Theory of Computing, STOC*, pages 580–589, 2001.

[11] Rosario Gennaro, Michael O. Rabin, and Tal Rabin. Simplified vss and fast-track multiparty computations with applications to threshold cryptography. In *17th annual ACM symposium on Principles of Distributed Computing, PODC*, pages 101–111, 1998.

[12] Amir Herzberg, Stanislaw Jarecki, Hugo Krawczyk, and Moti Yung. Proactive secret sharing or: How to cope with perpetual leakage. In *15th Annual International Cryptology Conference, CRYPTO*, volume 963 of *LNCS*, pages 339–352. Springer, 1995.

[13] Ingemar Ingemarsson and Gustavus J. Simmons. A protocol to set up shared secret schemes without the assistance of a mutualy trusted party. In Ivan Damgård, editor, *EUROCRYPT*, volume 473 of *Lecture Notes in Computer Science*, pages 266–282. Springer, 1990.

[14] Wen-Ai Jackson, Keith M. Martin, and Christine M. O'Keefe. Mutually trusted authority-free secret sharing schemes. *J. Cryptology*, 10(4):261–289, 1997.

[15] Ayako Maeda, Atsuko Miyaji, and Mitsuru Tada. Efficient and unconditionally secure verifiable threshold changeable scheme. In *6th Australasian Conference Information Security and Privacy, ACISP*, LNCS, pages 403–416. Springer, 2001.

[16] Keith Martin. Dynamic access policies for unconditionally secure secret sharing schemes. In *Proceedings of IEEE Information Theory Workshop (ITW 2005)*, pages 61–66. IEEE, 2005.

[17] Keith M. Martin, Josef Pieprzyk, Reihaneh Safavi-Naini, and Huaxiong Wang. Changing thresholds in the absence of secure channels. In *4th Australasian Conference Information Security and Privacy, ACISP*, volume 1587 of *LNCS*, pages 177–191. Springer, 1999.

[18] Keith M. Martin, Reihaneh Safavi-Naini, and Huaxiong Wang. Bounds and techniques for efficient redistribution of secret shares to new access structures. *The Computer Journal*, 42(8):638–649, 1999.

[19] Ventzislav Nikov and Svetla Nikova. On proactive secret sharing schemes. In *11th International Workshop on Selected Areas in Cryptography, SAC*, volume 3357 of *LNCS*, pages 308–325. Springer, 2004.

[20] M. Nojoumian, D.R. Stinson, and M. Grainger. Unconditionally secure social secret sharing scheme. *IET Information Security, Special Issue on Multi-Agent and Distributed Information Security*, 4(4):202–211, 2010.

[21] Tal Rabin and Michael Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *21st Annual ACM Symposium on Theory of Computing, STOC*, pages 73–85, 1989.

[22] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[23] Ron Steinfeld, Huaxiong Wang, and Josef Pieprzyk. Lattice-based threshold-changeability for standard shamir secret-sharing schemes. In *10th International Conference on the Theory and Application of Cryptology and Information Security, ASIACRYPT*, volume 3329 of *LNCS*, pages 170–186. Springer, 2004.

[24] Douglas R. Stinson and Ruizhong Wei. Unconditionally secure proactive secret sharing scheme with combinatorial structures. In *6th Annual Int. Workshop on Selected Areas in Cryptography, SAC*, volume 1758 of *LNCS*, pages 200–214. Springer, 1999.

[25] Christophe Tartary and Huaxiong Wang. Dynamic threshold and cheater resistance for shamir secret sharing scheme. In *2nd SKLOIS Conference on Information Security and Cryptology, Inscrypt*, volume 4318 of *LNCS*, pages 103–117. Springer, 2006.