Dynamic Threshold and Cheater Resistance for Shamir Secret Sharing Scheme*

Christophe Tartary[1] and Huaxiong Wang[1,2]

[1]Centre for Advanced Computing, Algorithms and Cryptography
Department of Computing
Macquarie University
NSW 2109 Australia

[2]Division of Mathematical Sciences
School of Physical and Mathematical Sciences
Nanyang Technological University
Singapore

ctartary@ics.mq.edu.au
HXWang@ntu.edu.sg

Abstract

In this paper, we investigate the problem of increasing the threshold parameter of the Shamir (t, n)-threshold scheme without interacting with the dealer. Our construction will reduce the problem of secret recovery to the polynomial reconstruction problem which can be solved using a recent algorithm by Guruswami and Sudan.
In addition to be dealer-free, our protocol does not increase the communication cost between the dealer and the n participants when compared to the original (t, n)-threshold scheme. Despite an increase of the asymptotic time complexity at the combiner, we show that recovering the secret from the output of the previous polynomial reconstruction algorithm is still realistic even for large values of t. Furthermore the scheme does not require every share to be authenticated before being processed by the combiner. This will enable us to reduce the number of elements to be publicly known to recover the secret to one digest produced by a collision resistant hash function which is smaller than the requirements of most verifiable secret sharing schemes.

Keywords: Secret Sharing Scheme, Polynomial Reconstruction Problem, Threshold Changeability, Insecure Network, Cheater Resistance.

1 Introduction

A *(t, n)-threshold secret sharing scheme* enables an authority called *dealer* to distribute a *secret* s as *shares* amongst n participants in such a way that any group of minimum size t can recover s while no groups having at most $t-1$ members can get any information about s. The recovery process is executed by an authority called *combiner*. When introduced in 1979 by Shamir [33] and Blakley [1], secret sharing was designed to facilitate the distributed storage of a secret s in an unreliable environment. Nowadays secret sharing protocols play an important role in group-oriented cryptography [5]. In such a context, it is likely to have an attacker trying to recover the secret value s. In addition, the attacker capabilities are likely to change over time requiring the threshold parameter t to be modified. Unfortunately, establishing secure communication between the n participants and the dealer to modify the threshold may not be possible after the original setup stage. Ideally, the dealer should not be required to take part to the threshold update process. A scheme having this property is said to provide *dealer-free* threshold changeability. Several such schemes have been designed but either they have large storage/communication requirements for the group members or the original (t, n)-scheme was specially designed for particular future threshold modification [3, 20, 21, 22]. Some constructions require communication between participants to achieve the threshold update [6, 23]. The recent construction by Steinfeld *et al.* [35] overcome these drawbacks provided that all participants are honest when recovering the secret s.

In a group-oriented context, it is also important to consider the case when some participants (or an outsider impersonating one of the participants) submit invalid shares to the combiner either to get some information about s or to prevent s from being reconstructed by the combiner. An approach to deal with an enemy $\mathcal{E}$ would be to let each participant flip a coin and send to the combiner either his share or an erasure according to the draw. Nevertheless, it is easy for $\mathcal{E}$ to differentiable an erasure from a non-erased symbol. Encrypting information would prevent $\mathcal{E}$ from getting knowledge of the transmitted

*The original version of this paper appears in the proceedings of the 2nd SKLOIS Conference on Information Security and Cryptology (INSCRYPT 2006), Lecture Notes in Computer Science, vol. 4318, pp 103 - 117, Springer - Verlag.

element. Nonetheless, encryption would not make the combiner immune against a substitution attack performed by $\mathcal{E}$ since $\mathcal{E}$ can substitute the eavesdropped value v_i by any random element v_i' uniformly chosen from $\mathbb{F}_p \setminus \{v_i\}$ (where $\mathbb{F}_p$ denote the finite field of prime order p). The probability that v_i' is the encryption of neither an erasure nor the share s_i is at least $1 - \frac{3}{p}$ which means that the element v_i', reaching the combiner, will be incorrect with high probability. Thus, recovery of s is not guaranteed by this approach (see Section 2). In [24], McEliece and Sarwate studied the case where some shares submitted to the combiner were faulty. The drawback of this approach is that the share construction depends on the number of incorrect values to be tolerated. Thus, the scheme is not secure if the abilities of $\mathcal{E}$ increase over time as it is likely to occur as said in earlier. The problem of dealing with dishonest participants was further studied in [9, 10]. Tompa and Woll [40] designed a generic attack on any linear secret sharing scheme enabling a dishonest participant to recover the value s while honest participants receive an invalid secret $\tilde{s} \neq s$ from the combiner. As noted in [41], there are two main ways of dealing with cheaters. The first approach is to discourage cheaters from sending incorrect shares to the combiner by designing the scheme in such a way that the cheater cannot recover s from the incorrect secret $\tilde{s}$ built by the combiner. Although the protocols in [29, 27, 28, 41] are unconditionally secure, they do not prevent the combiner from reconstructing an invalid secret $\tilde{s}$ when some participants submit incorrect shares. The second approach is to build the shares in such a way that the participants and/or the combiner can verify their correctness. Such schemes are called *Verifiable Secret Sharing* (VSS) schemes. Unfortunately, most VSS have requirements which become prohibitive when the group size n or the secret size $|s|$ gets large. For instance in [38], sharing integrity is verified using an interactive protocol between the dealer and each participant. In [18], the dealer has to create n additional elements and transmit each of them to every participant which involves a large communication overhead as well. In [32], the dealer has to publish $n+1$ elements of size $|s|$ and $n+t$ elements from a group of order $|s|$. The scheme in [8] is unconditionally secure but the dealer needs to distribute n polynomials of degree at most $t-1$ which represent a total of $n\,t$ extra elements of size $|s|$. Recently, Stinson and Zhang designed a technique dealing with up to ℓ cheaters [37]. Nevertheless, when there are exactly ℓ cheaters their Generalized Combined Algorithm will require all the n shares to have been submitted to the combiner in order to successfully recover s.

In this paper, we propose a computationally secure approach to modify dynamically the threshold of Shamir (t, n)-scheme according to the enemy's abilities. In our construction, each participant will receive one share from the dealer and will generate some random elements when sending data to the combiner. These random elements will first allow us to increase the threshold t to T by reducing the secret recovery problem to the polynomial reconstruction problem for which an efficient algorithm has been developed by Guruswami and Sudan [13]. We will show that it is computationally prohibitive to recover s from $T(\geq t)$ "shares" that are submitted to the combiner. Furthermore when up to $\mathcal{F}$ communication channels are corrupted either by an outsider or by dishonest participants, Guruswami and Sudan's algorithm will enable us to deduce another threshold $T_{\mathcal{F}}$ from which secret recovery is guaranteed. In addition to be dealer-free, our protocol has several advantages over the previous constructions. First, the communication cost between the dealer and the n participant will be identical to the cost of the original Shamir (t, n)-threshold scheme and therefore it will be independent from both n and t. Second, a single element will need to be publicly known to recover s. Finally, exhausting the output of the polynomial reconstruction algorithm to recover s will remain practical even for large values of t.

This paper is organized as follows. In the next section, we will recall the construction of Shamir secret sharing scheme and introduce two standard polynomial reconstruction problems. They will be used to design and prove the security and soundness of our construction in Section 3. In Section 4, we will study the practical efficiency of our scheme over an insecure network. In the last section, we will summarize the benefits of our protocol to the problem of dealing with enemies in secret sharing schemes over an insecure environment.

2 Preliminaries

2.1 Shamir Secret Sharing Scheme

Let p be a prime number. Denote $s \in \mathbb{F}_p$ the secret to be shared amongst the n participants $P_1, \ldots, P_n$. Every participant P_i is given a unique identification value $x_i \in \mathbb{F}_p^*$.

Share Construction. The dealer uniformly selects $t-1$ coefficients $a_1, \ldots, a_{t-1}$ from $\mathbb{F}_p$ and builds the polynomial $S(X)$ over $\mathbb{F}_p$ as:

$$S(X) := s + \sum_{i=1}^{t-1} a_i\, X^i$$

and computes the n shares $(x_1, y_1), \ldots, (x_n, y_n)$ as: $\forall i \in \{1, \ldots, n\}\ y_i := S(x_i)$. Share (x_i, y_i) is given to participant P_i for i in $\{1, \ldots, n\}$.

Secret Recovery. Assume that the combiner receives t shares $(x_{i_1}, y_{i_1}), \ldots, (x_{i_t}, y_{i_t})$. He reconstructs the polynomial

$S(X)$ using Lagrange interpolation formula as:

$$S(X) = \sum_{j=1}^{t} y_{i_j} \left(\prod_{\substack{1 \le k \le t \\ k \neq j}} \frac{X - x_{i_k}}{x_{i_j} - x_{i_k}} \right)$$

and recovers s since: $s = S(0)$.

Cheating in Shamir Secret Sharing Scheme. Assume that at least one of the participants (say P_{i_j}) sends an incorrect share to the combiner. The incorrect share can be written as $(x_{i_j}, \tilde{y}_{i_j})$ where $\tilde{y}_{i_j} \neq y_{i_j}$. Denote $\tilde{S}(X)$ the polynomial obtained by the combiner. Since $\tilde{S}(X)$ is a univariate polynomial passing through $(x_{i_j}, \tilde{y}_{i_j})$ we have $\tilde{S}(X) \neq S(X)$. Therefore, it is unlikely to have $s = \tilde{S}(0)$. Thus secret recovery cannot be guaranteed even if there is only one dishonest participant.

In addition, any outsider to the group $P_1, \ldots, P_n$ can recover s by eavesdropping the communication between t participants and the combiner using the same reconstruction process as the combiner.

2.2 Reconstructing Polynomials

We introduce two computational problems related to the reconstruction of polynomials.

Polynomial Reconstruction Problem (PRP). Given as input K, T and N points $(x_1, y_1), \ldots, (x_N, y_N)$ from $\mathbb{F}_q^2$ (where q is the power of a prime number p), output all univariate polynomials $P(X)$ of degree at most K such that $y_i = P(x_i)$ for at least T values of i.

In [13], Guruswami and Sudan developed an algorithm called Poly-Reconstruct outputting such a list provided that: $T \geq \sqrt{(1+\epsilon)\, K\, N}$ for some positive constant ϵ. It requires $O\left(N^2\, \epsilon^{-5}\, \log q\right)$ field operations in $\mathbb{F}_q$ and the list has size $O\left(\epsilon^{-5} \sqrt{\frac{N}{T}}\right)$ [12]. Therefore, in order to use Poly-Reconstruct, we must have $T > \sqrt{K\, N}$. Poly-Reconstruct is the best known algorithm to solve PRP [2].

Noisy Polynomial Interpolation Problem (NPIP). Let $P(X)$ be a polynomial of degree K over the finite field $\mathbb{F}_q^2$ (where q is the power of a prime number p). Given $N > K + 1$ sets $S_1, \ldots, S_N$ and N distinct field elements $x_1, \ldots, x_N$ such each $S_i := \{y_{i1}, \ldots, y_{im}\}$ contains $m-1$ random elements and $P(x_i)$, recover $P(X)$ provided that the solution is unique.

It is clear that Poly-Reconstruct can be used to solve NPIP when $N > \sqrt{K\, N\, m}$ (i.e. $N > K\, m$). In [2], Bleichenbacher and Nguyen studied the case when $N \leq K\, m$. They enlightened that when the size of the field characteristic $|p|$ was at least 80 bits long, the previously known techniques solving NPIP were computationally impractical while their construction based on lattices was efficient for K up to 154 and $m = 2$. Nevertheless, they pointed out two important facts. First they could not guarantee that the value $K = 155$ was reachable in practice (when $m = 2$) and second, their algorithm took one day to recover $P(X)$ when $m = 3$ for K as small as 101 on a 500 MHz 64-bit DEC Alpha using Victor Shoup's NTL Library (version 3.7a) [34] and Schnorr's BKZ reduction [30, 31].

3 Our Construction

As in [2], we consider that the size of the secret is at least 80 bits long. This is a realistic asumption since secret sharing protocols can be used to distribute keys for digital signatures (see [14] for instance). As in [15], the security of our construction will rely on the difficulty of solving NPIP. As said in Section 1, the size of the group is considered to be large. Therefore, we can assume that the threshold t is beyond 160 for which Bleichenbacher and Nguyen's construction cannot be used to solve NPIP. This assumption involves a large number of participants n. An illustration of such a setting is electronic legislature [7, 11] where n can be of the order of hundreds. One can even expect an organization such as the United Nations to have committees of thousands of representatives. In our construction we also consider that the dealer is honest.

3.1 Distributing Shares

We need a collision resistant hash function h [26]. Let p be a prime number. Denote $s \in \mathbb{F}_p$ the secret to be shared amongst the n participants $P_1, \ldots, P_n$. Every participant P_i is given a unique identification value $x_i \in \mathbb{F}_p^*$.

Share Construction. The dealer uniformly selects $t-1$ coefficients $a_1, \ldots, a_{t-1}$ from $\mathbb{F}_p$ and builds the polynomial $S(X)$ over $\mathbb{F}_p$ as:

$$S(X) := s + \sum_{i=1}^{t-1} a_i X^i$$

He computes the n shares $y_1, \ldots, y_n$ as: $\forall i \in \{1, \ldots, n\}$ $y_i := S(x_i)$ and publishes $h(s\|a_0\| \cdots \|a_{t-1})$. Share (x_i, y_i) is given to participant P_i over a secure channel for i in $\{1, \ldots, n\}$.

3.2 Threshold Increase and Secret Recovery

As said in Section 1, we will assume that the network (i.e. the communication channel between the participants and the combiner) is under control of an outsider who can eavesdrop communications. If the (t, n)-scheme has to be used over a long period of time then it is likely the adversary increases his computational power and therefore threatens the security of the scheme. As a consequence, the threshold value t may need to be increased to preserve the secret s from being recovered by an enemy eavesdropping the network. Nevertheless, the dealer may not be part of the network when the threshold increase is needed. As in [35], our construction will be dealer-free and the new threshold value will be set by the combiner. It can be reliably transmitted to each participant over the insecure channel using a digital signature [36].

Let λ be any positive integer such that: $n \geq (t-1)\,\lambda + 1$. Assume that each participant P_i randomly chooses $\lambda - 1$ distinct elements from $\mathbb{F}_p \setminus \{y_i\}$ and sends the whole set of λ coefficients to the combiner. Upon reception of data from t' participants, the combiner has to solve an instance of NPIP. As said in Section 1, Poly-Reconstruct can be used provided:

$$t' > \sqrt{(t-1)\,(\lambda\, t')}$$

which is equivalent to: $t' > (t-1)\,\lambda$. As said in Section 2, the list output by Poly-Reconstruct contains the polynomial $S(X)$. Denote T the integer defined as: $T := (t-1)\,\lambda + 1$. Notice that: $T \geq t$.

Combining Shares. Assume that the combiner receives $\lambda\, T$ elements from T participants. He runs Poly-Reconstruct to obtain a list of candidates for the polynomial $S(X)$. Using the public value $h(s\|a_1\| \cdots \|a_{t-1})$, the receiver recovers $S(X)$ from the list (as in [17]) and obtains s as: $s = S(0)$. Notice that the use of h is at the cost of losing information theoretic security. As we will see later, computational security will be guaranteed by our choice of t.

Remark, when $\lambda = 1$, our construction is identical to Shamir (t, n)-threshold scheme. Without loss of generality, we will assume that $\lambda \geq 2$ is the remaining of this paper.

3.3 Dealing with Eavesdroppers

3.3.1 Passive Eavesdropper

We first consider that the eavesdropper $\mathcal{E}$ is *passive*. This means that $\mathcal{E}$ can read information but cannot modify or inject data into the network. It should be noticed that if he can spy T communication channels then he can recover s in the same way as the combiner since the value $h(s\|a_1\| \cdots \|a_{t-1})$ is public. Now, we have to argue that if $\mathcal{E}$ has access to at most $T-1$ channels then it is computationally impossible for him to recover s. Denote $\mathcal{C}$ the number of eavesdropped channels and $P_{i_1}, \cdots, P_{i_\mathcal{C}}$ the corresponding participants. Since $\mathcal{C} < T$, $\mathcal{E}$ cannot use Poly-Reconstruct on the $\lambda\,\mathcal{C}$ values he obtained to recover $S(X)$. We have three cases to consider.

First case: $\mathcal{C} \geq t+1$
Since at least $t+1$ participants are eavesdropped, the problem of recovering $S(X)$ from the $\lambda\,\mathcal{C}$ points is an instance of NPIP. Due to our choice of parameters p and t no algorithm is known to solve efficiently NPIP (see Section 2).

Nevertheless, data from any t participants $P_{i_1}, \ldots, P_{i_t}$ can still be used to reconstruct polynomials of degree at most $t-1$ using Lagrange interpolation formula by picking one elements from each participant. There is a total of λ^t polynomials including $S(X)$. For each of these polynomials one hash is computed and compared to the public value $h(s\|a_1\| \cdots \|a_{t-1})$. The average number of hashes of $(t \log_2 p)$-bit messages to be computed is:

$$\frac{\lambda^t + 1}{2}$$

This approach becomes computationally prohibitive since $\lambda \geq 2$ and $t \geq 160$.

Second case: $\mathcal{C} = t$
According to Section 2.2, NPIP is defined provided that we have $N > K + 1$ sets of elements (where K is the degree of the polynomial). In our case, this inequality can be written as: $\mathcal{C} > t$. This means that attempting to solve NPIP when

$\mathcal{C} = t$ is irrelevant. Therefore, the only known approach $\mathcal{E}$ can follow is to use Lagrange interpolation formula as in the first case.

Third case: $\mathcal{C} < t$
In this case, Lagrange interpolation formula can only be used to obtain polynomials of degree at most $\mathcal{C} - 1$. Nevertheless, the probability that the degree of $S(X)$ is at most $\mathcal{C} - 1$ is extremely small. Indeed, using the fact the $a_1, \ldots, a_{t-1}$ are drawn uniformly at random from $\mathbb{F}_p$ we have:

$$\text{prob}(\deg(S(X)) \leq \mathcal{C} - 1) = \text{prob}(a_{\mathcal{C}} = \cdots = a_{t-1} = 0) = \frac{1}{p^{t-1-\mathcal{C}}}$$

Since p is at least 80 bits long, we have: $p > 2^{79}$. We obtain:

$$\text{prob}(\deg(S(X)) \leq \mathcal{C} - 1) \leq 2^{-79(t-\mathcal{C})}$$

where: $t - \mathcal{C} \geq 1$. In particular we get:

$$\text{prob}(\deg(S(X) = t - 1)) \geq 1 - \frac{1}{2^{79}} \geq 1 - 10^{-21}$$

Thus, the only approach $\mathcal{E}$ can use is to solve linear systems of t unknowns and $\mathcal{C}$ equations. This yields to a total of $\lambda^{\mathcal{C}}\, p^{t-\mathcal{C}}$ polynomials including $S(X)$. The average number of hashes of $(t \log_2 p)$-bit messages to be computed is:

$$\frac{\lambda^{\mathcal{C}}\, p^{t-\mathcal{C}} + 1}{2}$$

Since the field $\mathbb{F}_p$ has p elements, we have: $\lambda \leq p$. Therefore, the previous value is lower bounded by $\frac{\lambda^t+1}{2}$ which represents the complexity of the first two cases.

We deduce that if $\mathcal{E}$ cannot eavesdrop more than $T-1$ channels then it is computationally impossible for him to recover the secret s. Thus the threshold value of Shamir (t, n)-threshold scheme set by the dealer has been increased to T where $T = (t-1)\,\lambda + 1$. We have:

$$\frac{T}{t} = \left(1 - \frac{1}{t}\right)\lambda + \frac{1}{t} \simeq \lambda$$

Thus, the original threshold value t has been approximately multiplied by λ (without the intervention of the dealer).

3.3.2 Active Eavesdropper

We now assume that the eavesdropper $\mathcal{E}$ is *active*. This means that $\mathcal{E}$ can also modify or inject data into the network. His goal is to prevent the participants to recover the secret s. He has two ways to proceed:

First Case. $\mathcal{E}$ can drop or inject data such that the combiner receives either two identical elements or $\tilde{\lambda}_i \neq \lambda$ elements from some participant P_i. In this case, the combiner can simply ignore data originating from P_i when running Poly-Reconstruct. If $\mathcal{E}$ performs this action on several channels then it may result in a denial-of-service attack on the secret reconstruction process. Nevertheless, this attack has two drawbacks for $\mathcal{E}$. First, it will not result in an incorrect secret $\tilde{s}$ to be recovered by the combiner (contrary to [40]). Second, it reveals the presence of $\mathcal{E}$ to the combiner. Thus, the latter can decide to set up another communication channel between P_i and himself which is likely not to be under control of $\mathcal{E}$. Therefore, we will not consider that $\mathcal{E}$ performs this kind of attack any longer.

Second Case. $\mathcal{E}$ can substitute data originating from P_i by his own forgery in such a way that the combiner receives exactly λ distinct elements. The forgery is successful if the share y_i is not amongst the set of elements received at the combiner. To achieve this goal, $\mathcal{E}$ has to generate λ new distinct elements from $\mathbb{F}_p$ since y_i is indistinguishable from the remaining $\lambda - 1$ elements generated by P_i.

Assume the combiner receives data from t' participants where up to $\mathcal{C}$ channels have been corrupted by $\mathcal{E}$. The combiner must solve an instance of PRP. Consider: $n \geq \sqrt{(t-1)\,\lambda\, n} + 1 + \mathcal{C}$. The combiner can use Poly-Reconstruct provided:

$$t' - \mathcal{C} > \sqrt{(t-1)\,\lambda\, t'}$$

Since $n \geq \sqrt{(t-1)\,\lambda\, n} + 1 + \mathcal{C}$, we can define the integer $T_{\mathcal{C}}$ as:

$$T_{\mathcal{C}} := \min\left\{t' \in \{1, \ldots, n\} \,|\, t' - \mathcal{C} > \sqrt{(t-1)\,\lambda\, t'}\right\}$$

Therefore, it is sufficient for the combiner to collect data from $T_{\mathcal{C}}$ participants to run Poly-Reconstruct and recover s. More details concerning the value $T_{\mathcal{C}}$ and the size of the list output by Poly-Reconstruct will be given in Section 3.5.

It can be noticed that we can deal with active eavesdroppers using authentication protocols for insecure channels such as [16, 19, 39]. They prevent forgeries to be accepted by the combiner. In addition, the use of an error correcting code [16] will also allow the combiner to recover from element erasures. In those protocols, the role of the sender will be played by each participant in turn whereas the receiver will be the combiner.

3.4 Dealing with Dishonest Participants

The recovery process from Section 3.2 is guaranteed to work if every participant is honest as in [35]. This means that each participant submits to the combiner λ distinct elements including his share y_i. Nevertheless the larger the group is, the higher the probability of having dishonest participants becomes.

A dishonest participant P_i can act in two different ways. First, he can submit either two identical elements or $\tilde{\lambda}_i \neq \lambda$ elements to the combiner. Second, he does not submit his correct share y_i. The first case is ruled out in the same way as in the case of an active eavesdropper in Section 3.3 except that the combiner does not set up any other channel to P_i but removes P_i from the group of trusted members. The second case can be treated as in Section 3.3 as well. Namely, we assume the combiner receives data from t' participants where up to $\mathcal{D}$ are dishonest. The combiner must solve an instance of PRP. Consider: $n \geq \sqrt{(t-1)\,\lambda\,n} + 1 + \mathcal{D}$. The combiner can use Poly-Reconstruct provided:

$$t' - \mathcal{D} > \sqrt{(t-1)\,\lambda\,t'}$$

Since $n \geq \sqrt{(t-1)\,\lambda\,n} + 1 + \mathcal{D}$, we can define the integer $T_{\mathcal{D}}$ as:

$$T_{\mathcal{D}} := \min\left\{t' \in \{1,\ldots,n\} \,|\, t' - \mathcal{D} > \sqrt{(t-1)\,\lambda\,t'}\right\}$$

Therefore, it is sufficient for the combiner to collect data from $T_{\mathcal{D}}$ participants to run Poly-Reconstruct and recover s. More details concerning the value $T_{\mathcal{D}}$ and the size of the list output by Poly-Reconstruct will be given in Section 3.5.

3.5 Dealing with Invalid Data

In this section, we will combine the results from Section 3.3 and Section 3.4. We consider that there are at most $\mathcal{D}$ dishonest participants amongst the group of n members and at most $\mathcal{C}$ channels are under control of an active eavesdropper $\mathcal{E}$. Even if $\mathcal{E}$ can modify data sent by a dishonest participant, we only consider the worst case where $\mathcal{E}$ only acts on channels between the combiner and honest participants. Therefore, the number of channels transferring invalid data is at most $\mathcal{D} + \mathcal{C}$. This bound is denoted $\mathcal{F}$. We assume the combiner receives data from t' participants where up to $\mathcal{F}$ channels are faulty. The combiner must solve an instance of PRP. Consider: $n \geq \sqrt{(t-1)\,\lambda\,n} + 1 + \mathcal{F}$. The combiner can use Poly-Reconstruct provided:

$$t' - \mathcal{F} > \sqrt{(t-1)\,\lambda\,t'}$$

Since $n \geq \sqrt{(t-1)\,\lambda\,n} + 1 + \mathcal{F}$, we can define the integer $T_{\mathcal{F}}$ as:

$$T_{\mathcal{F}} := \min\left\{t' \in \{1,\ldots,n\} \,|\, t' - \mathcal{F} > \sqrt{(t-1)\,\lambda\,t'}\right\}$$

We have the following theorem the proof of which can be found in Appendix A.

Theorem 1 *If up to $\mathcal{F}$ channels are faulty then the combiner needs to obtained the elements of (at least) $T_{\mathcal{F}}$ participants to run Poly-Reconstruct. The size of the output list is upper bounded by $U_{\mathcal{F}}$ and includes $S(X)$ where:*

$$T_{\mathcal{F}} = \begin{cases} \left\lceil \mathcal{F} + \frac{(t-1)\,\lambda\,\sqrt{\Delta_{\mathcal{F}}}}{2} \right\rceil & \text{if } \frac{(t-1)\,\lambda\,\sqrt{\Delta_{\mathcal{F}}}}{2} \notin \mathbb{N} \\ \mathcal{F} + 1 + \frac{(t-1)\,\lambda\,\sqrt{\Delta_{\mathcal{F}}}}{2} & \text{otherwise} \end{cases}$$

$$U_{\mathcal{F}} = \left\lfloor \frac{T_{\mathcal{F}} - \mathcal{F} - 1}{t-1} + \frac{T_{\mathcal{F}} - \mathcal{F}}{(T_{\mathcal{F}} - \mathcal{F})^2 - (t-1)\,\lambda\,T_{\mathcal{F}}} \left(\lambda\,T_{\mathcal{F}} + \frac{\sqrt{(T_{\mathcal{F}} - \mathcal{F})^2 - (t-1)\,\lambda\,T_{\mathcal{F}}}}{t-1} \right) \right\rfloor$$

with: $\Delta_{\mathcal{F}} = (4\,\mathcal{F} + (t-1)\,\lambda)\,(t-1)\,\lambda$.

The values of $T_{\mathcal{E}}$ and $T_{\mathcal{D}}$ from Section 3.3 and Section 3.4 can be obtained from Theorem 1 substituting $\mathcal{F}$ by $\mathcal{E}$ and $\mathcal{D}$. The size of the lists output by Poly-Reconstruct are bounded by $U_{\mathcal{E}}$ and $U_{\mathcal{D}}$ respectively.

4 Efficiency of Secret Recovery

In this section we will study the computational cost of our construction for the dealer and the combiner when up to $\mathcal{F}$ channels are faulty.

4.1 Computation Cost of the Protocol

Cost at the Dealer. The dealer needs to generates n shares as evaluations of a polynomial of degree at most $t-1$ over $\mathbb{F}_p$. Using Horner's method, the evaluation of a polynomial of degree d requires $d-1$ multiplications and $d-1$ additions in the field. In our case, we have: $d \leq t-1$. That is, the dealer performs at most $t-1$ additions and $t-1$ multiplications in $\mathbb{F}_p$ to build the n shares. This is the same cost as Shamir (t, n)-threshold scheme.

Cost at the Combiner. In Section 3.5, we showed that if the combiner receives data from $T_{\mathcal{F}}$ participants then he could recover $S(X)$ by exhausting the list output by Poly-Reconstruct the size of which is upper bounded by $U_{\mathcal{F}}$. As said in Section 2, the number of field operations to build the list of polynomials is $O((\lambda T_{\mathcal{F}})^2 \epsilon^{-5} \log_2 p)$ where $T_{\mathcal{F}} - \mathcal{F} \geq \sqrt{(1+\epsilon)(t-1)\lambda T_{\mathcal{F}}}$ and at most one hash is computed per list element.

Table 1 summarizes the cost of our construction. We did not include the cost of computing $S(0)$ since $S(0)$ is the lowest degree coefficient of $S(X)$.

Dealer	Combiner	Participant
$n(t-1)$ multiplications $n(t-1)$ additions	$O((\lambda T_{\mathcal{F}})^2 \epsilon^{-5} \log_2 p)$ field operations $U_{\mathcal{F}}$ hashes of $(t \log_2 p)$-bit messages	Storage of 1 element

Table 1: Cost of our construction when up to $\mathcal{F}$ channels are faulty.

4.2 Efficiency Comparison

As claimed in Section 1, each participant receives one share. This leads to a communication cost between the dealer and the n participants which is identical to the original Shamir (t, n)-threshold scheme. This constitutes a major advantage over constructions such as [8, 18] since the group size is assumed to be large. In addition, we only require one hash value to be publicly known to recover the secret s contrary to [32] where (at least) $n+1$ elements of size $|s|$ need to be known. Since the secret is at least 80 bits long, the construction by Schoenmakers involves $80(n+1)$ bits to be reserved in a public register. This is larger than one digest produced by SHA-256 for $n \geq 4$.

The time for recovering $S(X)$ from the list output by Poly-Reconstruct is an important parameter for the efficiency of our scheme. We illustrated the results from Theorem 1 using SHA-256 as a collision resistant hash function. We considered that p was 80 bits long. Based on Dai's benchmarks [4], our results are shown in Table 2 where the time unit for t_{SHA} is the second.

Contrary to Bleichenbacher and Nguyen's technique, Table 2 seems to indicate that for t up to 200 and even in the presence of up to 100 dishonest participants, recovering $S(X)$ from the list output by Poly-Reconstruct is quite fast (155 seconds) even for small values of λ. This behavior is confirmed for larger values of t since the list is exhausted in less than 36 minutes (2152 seconds) when $t = 500$.

The list decoding result by Guruswami and Sudan used in this paper was recently extended by Parvaresh and Vardy [25] where a larger number of incorrect symbols can be tolerated. Nevertheless, this improvement occurs only when the ratio $\frac{t}{n}$ does not exceed $\frac{1}{16}$. In addition, they did not provide an explicit decoding time complexity (despite it is in polynomial time) contrary to Guruswami and Sudan where the decoding time complexity of their algorithm is $O\left(n^2 \epsilon^{-5} \log^2 p \log^{O(1)} \log p\right)$ where ϵ is defined as in Section 2.2 [12].

4.3 Limitations

Our construction has two drawbacks related to the use of Poly-Reconstruct. Contrary to [35], we cannot choose our new threshold $T > t$ since this new value is fixed as $(t-1)\lambda + 1$. Second, in order to have $n \geq \sqrt{(t-1)\lambda n} + 1 + \mathcal{F}$ we must have:

$$t \geq \frac{n}{\lambda}\left(1 - \frac{1+\mathcal{F}}{n}\right) + 1$$

In particular, we must have $t \leq \frac{n}{2}$ since $\lambda \geq 2$. Therefore, our scheme is not constructible when t is larger than half the group size. So, our scheme is limited to large groups where the initial threshold is modest which enlightens the efficiency

t	λ	T	$\mathcal{F}$	$T_{\mathcal{F}}$	$U_{\mathcal{F}}$	t_{SHA}	t	λ	T	$\mathcal{F}$	$T_{\mathcal{F}}$	$U_{\mathcal{F}}$	t_{SHA}
160	2	319	10	338	2219	1	250	2	499	10	518	5265	3
			50	412	10655	3				50	594	5214	3
			100	498	9913	3				100	684	1886	1
	5	796	10	815	32809	10		5	1246	10	1265	79384	35
			50	893	5277	2				50	1344	7527	4
			100	985	29063	8				100	1439	7058	4
	7	1114	10	1133	89072	25		7	1744	10	1763	216345	94
			50	1211	126185	35				50	1842	35123	16
			100	1306	12857	4				100	1938	80441	35
	10	1591	10	1610	257611	71		10	2491	10	2510	627511	270
			50	1689	34144	10				50	2590	26324	12
			100	1785	27989	8				100	2687	35860	16
180	2	359	10	378	2784	1	300	2	599	10	618	7517	4
			50	453	1156	1				50	695	2162	2
			100	540	1699	1				100	786	1900	1
	5	896	10	915	41409	13		5	1496	10	1515	114009	59
			50	993	9114	3				50	1594	13587	7
			100	1086	23696	8				100	1690	8673	5
	7	1254	10	1273	112553	35		7	2094	10	2113	311062	161
			50	1352	10741	4				50	2192	106718	55
			100	1447	10359	4				100	2289	41564	22
	10	1791	10	1810	325811	101		10	2991	10	3010	903011	466
			50	1889	56866	18				50	3090	37584	20
			100	1985	498908	155				100	3187	224117	116
200	2	399	10	418	3413	2	500	2	999	10	1018	20525	18
			50	493	12482	5				50	1096	7446	7
			100	581	4546	2				100	1190	5406	5
	5	996	10	1015	51009	18		5	2496	10	2515	315009	271
			50	1093	18158	7				50	2595	13213	12
			100	1187	12805	5				100	2692	18138	16
	7	1394	10	1413	138778	48		7	3494	10	3513	861430	740
			50	1492	14948	6				50	3593	35651	31
			100	1587	34566	12				100	3691	35446	31
	10	1991	10	2010	402011	139		10	4991	10	5010	2505011	2152
			50	2089	103647	36				50	5090	102624	89
			100	2186	36316	13				100	5189	54898	48

Table 2: Benchmarks for our scheme when using SHA-256 as a hash function.

limitation of the algorithm by Guruswami and Sudan in the context of secret sharing.

We would also like to draw the reader's attention to the fact that if any t participants collude then they are able to recover $S(X)$ and then the secret s using Lagrange interpolation formula as in the (t, n)-threshold scheme. Nevertheless, the larger t is, the harder having secure channels between t participants gets.

5 Conclusion

We introduced a new approach to allow a flexible change for the threshold value of Shamir (t, n)-threshold scheme over an insecure network. Our construction does not require the dealer to take part in the update process. In addition, the communication cost between the dealer and the participants is identical to the original (t, n)-threshold scheme. Furthermore, the size of the data to be stored in a public register to ensure the recover of the secret is constant and therefore does not depend on the group size n. This is particularly valuable in the context of group oriented cryptography where the size of the group may be large. We also showed that the combiner could recover the secret s from the list output by Poly-Reconstruct within a reasonable time period even for large values of t. Nevertheless, we enlightened that this algorithm did not allow any value t' to be chosen as the new threshold (contrary to [35]) and required the original threshold t to be no larger than $\frac{n}{2}$.

Acknowledgment

The authors would like to thank Professor Josef Pieprzyk for valuable conversations about secret sharing schemes. We are also grateful to the anonymous reviewers for their comments to improve the quality of this paper. This work was supported by the Australian Research Council under ARC Discovery Projects DP0558773 and DP0665035. The first author's work was also funded by an iMURS scholarship supported by Macquarie University.

References

[1] George Robert Blakley. Safeguarding cryptographic keys. In *AFIPS 1979 National Computer Conference*, pages 313 – 317, New York, USA, June 1979. AFIPS Press.

[2] Daniel Bleichenbacher and Phong Q. Nguyen. Noisy polynomial interpolation and noisy chinese remaindering. In *Advances in Cryptology - Eurocrypt'00*, volume 1807 of *Lecture Notes in Computer Science*, pages 53 – 69, Bruges - Belgium, May 2000. Springer - Verlag.

[3] Carlo Blundo, Antonella Cresti, Alfredo De Santis, and Ugo Vaccaro. Fully dynamic secret sharing schemes. In *Advances in Cryptology - Crypto'93*, volume 773 of *Lecture Notes in Computer Science*, pages 110 – 125, Santa Barbara, USA, August 1994. Springer - Verlag.

[4] W. Dai. Crypto++ 5.2.1 benchmarks, July 2004.

[5] Yvo Desmedt. Society and group oriented cryptography: A new concept. In *Advances in Cryptology - Crypto'87*, volume 293 of *Lecture Notes in Computer Science*, pages 120 – 127, Santa Barbara, USA, August 1987. Springer - Verlag.

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

[7] Yvo Desmedt and Brian King. Verifiable democracy a protocol to secure an electronic legislature. In *EGOV 2002*, volume 2456 of *Lecture Notes in Computer Science*, pages 460 – 463, Aix-en-Provence, France, September 2002. Springer - Verlag.

[8] Yvo Desmedt, Kaoru Kurosawa, and Tri Van Le. Error correcting and complexity aspects of linear secret sharing schemes. In *6th International Conference on Information Security*, volume 2851 of *Lecture Notes in Computer Science*, pages 396 – 407, Bristol, United Kingdom, October 2003. Springer - Verlag.

[9] Yair Frankel, Peter Gemmel, Philip D. MacKenzie, and Moti Yung. Optimal-resilience proactive public-key cryptosystems. In *FOCS'97*, pages 384 – 393, Miami Beach, USA, October 1997. IEEE Press.

[10] Zvi Galil, Stuart Haber, and Moti Yung. Cryptographic computation: Secure fault-tolerant protocols and the public-key model (extended abstract). In *Advances in Cryptology - Crypto'87*, volume 293 of *Lecture Notes in Computer Science*, pages 135 – 155, Santa Barbara, USA, August 1987. Springer - Verlag.

[11] Hossein Ghodosi and Josef Pieprzyk. Democratic systems. In *ACISP 2001*, volume 2119 of *Lecture Notes in Computer Science*, pages 392 – 402, Sydney, Australia, July 2001. Springer - Verlag.

[12] Venkatesan Guruswami. *List Decoding of Error-Correcting Codes*. Springer-Verlag, 2004.

[13] Venkatesan Guruswami and Madhu Sudan. Improved decoding of Reed-Solomon and algebraic-geometric codes. *IEEE Transactions on Information Theory*, 45(6):1757 – 1767, September 1999.

[14] Lein Harn. Group-oriented (t, n)-threshold digital signature scheme and digital multisignature. *IEE Proceedings - Computers and Digital Techniques*, 141(5):307 – 313, September 1994.

[15] Ari Juels and Madhu Sudan. A fuzzy vault scheme. In *ISIT 2002*, page 408, Lausanne, Switzerland, July 2002. IEEE Press. Extended version avaliable at: `http://www.rsasecurity.com/rsalabs/staff/bios/ajuels/publications/fuzzy-vault/fuzzy_vault.pdf`.

[16] Chris Karlof, Naveen Sastry, Yaping Li, Adrian Perrig, and J. D. Tygar. Distillation codes and applications to DoS resistant multicast authentication. In *11th Network and Distributed Systems Security Symposium (NDSS)*, San Diego, USA, February 2004.

[17] Ehud D. Karnin, Jonathan W. Greene, and Martin E. Hellman. On secret sharing systems. *IEEE Transactions on Information Theory*, 29(1):35 – 41, January 1983.

[18] Qiong Li, Zhifang Wang, Xiamu Niu, and Shenghe Sun. A non-interactive modular verifiable secret sharing scheme. In *International Conference on Communications, Circuits and Systems*, pages 84 – 87, HongKong, China, May 2005. IEEE Press.

[19] Anna Lysyanskaya, Roberto Tamassia, and Nikos Triandopoulos. Multicast authentication in fully adversarial networks. In *IEEE Symposium on Security and Privacy*, pages 241 – 253, Oakland, USA, May 2003. IEEE Press.

[20] Ayako Maeda, Atsuko Miyaji, and Mitsuru Tada. Efficient and unconditionally secure verifiable threshold changeable scheme. In *ACISP 2001*, volume 2119 of *LNCS*, pages 402 – 416, Sydney, Australia, July 2001. Springer - Verlag.

[21] Keith Martin. Untrustworthy participants in secret sharing schemes. In *Cryptography and Coding III*, volume 45, pages 255 – 264. Oxford University Press, 1993.

[22] Keith M. Martin, Josef Pieprzyk, Rei Safavi-Naini, and Huaxiong Wang. Changing thresholds in the absence of secure channels. *Australian Computer Journal*, 31:34 – 43, 1999.

[23] Keith M. Martin, Rei 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, September 1999.

[24] Robert J. McEliece and Dilip V. Sarwate. On sharing secrets and Reed-Solomon codes. *Communications of the ACM*, 24(9):583 – 584, September 1981.

[25] Farzad Parvaresh and Alexander Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *46th Annual IEEE Symposium on Foundations of Computer Science*, pages 285 – 294, Pittsburgh, USA, October 2005. IEEE Computer Society.

[26] Josef Pieprzyk, Thomas Hardjono, and Jennifer Seberry. *Fundamentals of Computer Security*. Springer, 2003.

[27] Josef Pieprzyk and Xian-Mo Zhang. Cheating prevention in secret sharing over GF(p^t). In *Progress in Cryptology - Indocrypt 2001*, volume 2247 of *Lecture Notes in Computer Science*, pages 79 – 90, Chennai, India, December 2001. Springer - Verlag.

[28] Josef Pieprzyk and Xian-Mo Zhang. Constructions of cheating immune secret sharing. In *ICISC 2001*, volume 2288 of *Lecture Notes in Computer Science*, pages 261 – 278, Seoul, Korea, December 2001. Springer - Verlag.

[29] Josef Pieprzyk and Xian-Mo Zhang. On cheating immune secret sharing. *Discrete Mathematics and Theoretical Computer Science*, 6:253 – 264, March 2004.

[30] C. P. Schnorr. A hierarchy of polynomial lattice basis reduction algorithms. *Theoretical Computer Science*, 53:201 – 224, 1987.

[31] C. P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problem. *Mathematical Programming*, 66(1 - 3):181 – 199, August 1994.

[32] Berry Schoenmakers. A simple publicly verifiable secret sharing scheme and its application to electronic voting. In *Advances in Cryptology - Crypto'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 148 – 164, Santa Barbara, USA, August 1999. Springer - Verlag.

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

[34] V. Shoup. Number Theory Library (NTL). Available online at: `http://www.shoup.net/ntl/`.

[35] Ron Steinfeld, Huaxiong Wang, and Josef Pieprzyk. Lattice-based threshold-changeability for standard Shamir secret-sharing schemes. In *Advances in Cryptology - Asiacrypt'04*, volume 3329 of *Lecture Notes in Computer Science*, pages 170 – 186, Jeju Island, Korea, December 2004. Springer - Verlag.

[36] Douglas R. Stinson. *Cryptography: Theory and Practice (Third Edition)*. Chapman & Hall/CRC, 2006.

[37] Douglas R. Stinson and Sheng Zhang. Algorithms for detecting cheaters threshold schemes. To appear in the Journal of Combinatorial Mathematics and Combinatorial Computing. Available online at: `http://www.cacr.math.uwaterloo.ca/~dstinson/papers/cheat.pdf`, January 2006.

[38] Chunming Tang, Zhuojun Liu, and Mingsheng Wang. A verifiable secret sharing scheme with statistical zero-knowledge. Cryptology ePrint Archive, Report 2003/222, October 2003. Available online at: `http://eprint.iacr.org/2003/222.pdf`.

[39] Christophe Tartary and Huaxiong Wang. Efficient multicast stream authentication for the fully adversarial network. In *6th International Workshop on Information Security Applications*, volume 3786 of *Lecture Notes in Computer Science*, pages 108 – 125, Jeju Island, Korea, August 2005. Springer - Verlag.

[40] M. Tompa and H. Woll. How to share a secret with cheaters. In *Advances in Cryptology - Crypto'86*, volume 263 of *Lecture Notes in Computer Science*, pages 261 – 265, Santa Barbara, USA, August 1986. Springer - Verlag.

[41] Xian-Mo Zhang and Josef Pieprzyk. Cheating immune secret sharing. In *ICICS 2001*, volume 2229 of *Lecture Notes in Computer Science*, pages 144 – 149, Xian, China, November 2001. Springer - Verlag.

A Proof of Theorem 1

Determination of $T_{\mathcal{F}}$. We are interested in determining the smallest positive integer $T_{\mathcal{F}}$ such that if up to $\mathcal{F}$ channels are faulty, the combiner can run Poly-Reconstruct after collecting the elements of $T_{\mathcal{F}}$ members. Since each participant has λ elements and the combiner obtain correct data from at least $T_{\mathcal{F}} - \mathcal{F}$ of them, the value $T_{\mathcal{F}}$ is the smallest positive integer $\mathcal{T}$ such that:

$$\mathcal{T} - \mathcal{F} > \sqrt{(t-1)\,\lambda\,\mathcal{T}} \tag{1}$$

Since $T_{\mathcal{F}}$ verifies Inequlity (1), we deduce: $T_{\mathcal{F}}^2 - (2\,\mathcal{F} + (t-1)\,\lambda)\,T_{\mathcal{F}} + \mathcal{F}^2 > 0$. Since the polynomial $X^2 - (2\,\mathcal{F} + (t-1)\,\lambda)\,X + \mathcal{F}^2$ has a positive discriminant: $\Delta_{\mathcal{F}} = (4\,\mathcal{F} + (t-1)\,\lambda)\,(t-1)\,\lambda$ we deduce that its two roots are:

$$r_1 := \mathcal{F} + \frac{(t-1)\,\lambda - \sqrt{\Delta_{\mathcal{F}}}}{2} \quad \text{and} \quad r_2 := \mathcal{F} + \frac{(t-1)\,\lambda + \sqrt{\Delta_{\mathcal{F}}}}{2}$$

We have: $r_1 < \mathcal{F} < r_2$ since $(t-1)\,\lambda < \sqrt{\Delta_{\mathcal{F}}}$. Since $T_{\mathcal{F}}$ has to verify Inequlity (1), we must have: $T_{\mathcal{F}} > \mathcal{F}$. Therefore: $T_{\mathcal{F}} \geq \tilde{T}_{\mathcal{F}}$ where:

$$\tilde{T}_{\mathcal{F}} = \begin{cases} \lceil r_2 \rceil & \text{if } r_2 \notin \mathbb{N} \\ r_2 + 1 & \text{otherwise} \end{cases}$$

By definition of $\tilde{T}_{\mathcal{F}}$, we have: $\tilde{T}_{\mathcal{F}}^2 - (2\,\mathcal{F} + (t-1)\,\lambda)\,\tilde{T}_{\mathcal{F}} + \mathcal{F}^2 > 0$ which is equivalent to:

$$|\tilde{T}_{\mathcal{F}} - \mathcal{F}| > \sqrt{(t-1)\,\lambda\,\tilde{T}_{\mathcal{F}}}$$

Since $\tilde{T}_{\mathcal{F}} > \mathcal{F}$, we deduce that $\tilde{T}_{\mathcal{F}}$ verifies Inequlity (1). Due to the minimality of $T_{\mathcal{F}}$ we obtain: $T_{\mathcal{F}} = \tilde{T}_{\mathcal{F}}$.

Determination of $U_{\mathcal{F}}$. We have at least $T_{\mathcal{F}} - \mathcal{F}$ correct shares amongst a total of $\lambda\,T_{\mathcal{F}}$ elements. Using Proposition 6.15 from [12], the size of the list output by Poly-Reconstruct is upper bounded by $\lfloor \frac{L}{t-1} \rfloor$ where:

$$\begin{cases} L := R\,(T_{\mathcal{F}} - \mathcal{F}) - 1 \\ \\ R := 1 + \left\lfloor \frac{(t-1)\,\lambda\,T_{\mathcal{F}} + \sqrt{((t-1)\,\lambda\,T_{\mathcal{F}})^2 + 4\,((T_{\mathcal{F}} - \mathcal{F})^2 - (t-1)\,\lambda\,T_{\mathcal{F}})}}{2\,((T_{\mathcal{F}} - \mathcal{F})^2 - (t-1)\,\lambda\,T_{\mathcal{F}})} \right\rfloor \end{cases}$$

We have the property: $\forall (a,b) \in \mathbb{R}^+ \times \mathbb{R}^+ \; \sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$. We obtain:

$$R \leq 1 + \frac{(t-1)\,\lambda\,T_{\mathcal{F}} + \sqrt{(T_{\mathcal{F}} - \mathcal{F})^2 - (t-1)\,\lambda\,T_{\mathcal{F}}}}{(T_{\mathcal{F}} - \mathcal{F})^2 - (t-1)\,\lambda\,T_{\mathcal{F}}}$$

From the definition of L and using the previous inequality as well as the increase of the function $x \mapsto \lfloor x \rfloor$ over its domain, we deduce:

$$\left\lfloor \frac{L}{t-1} \right\rfloor \leq \left\lfloor \frac{T_{\mathcal{F}} - \mathcal{F} - 1}{t-1} + \frac{T_{\mathcal{F}} - \mathcal{F}}{(T_{\mathcal{F}} - \mathcal{F})^2 - (t-1)\,\lambda\, T_{\mathcal{F}}} \left(\lambda\, T_{\mathcal{F}} + \frac{\sqrt{(T_{\mathcal{F}} - \mathcal{F})^2 - (t-1)\,\lambda\, T_{\mathcal{F}}}}{t-1} \right) \right\rfloor$$

Notice that the right hand side of the inequality is the value $U_{\mathcal{F}}$ defined in Theorem 1.