Verifiable Multi-secret Sharing Schemes for Multiple Threshold Access Structures

Christophe Tartary[1,2], Josef Pieprzyk[3], and Huaxiong Wang[1,3]

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

[2] Institute for Theoretical Computer Science
Tsinghua University
Beijing, 100084
People's Republic of China

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

{ctartary,josef}@ics.mq.edu.au
HXWang@ntu.edu.sg

Abstract. A multi-secret sharing scheme allows several secrets to be shared amongst a group of participants. In 2005, Shao and Cao developed a verifiable multi-secret sharing scheme where each participant's share can be used several times which reduces the number of interactions between the dealer and the group members. In addition some secrets may require a higher security level than others involving the need for different threshold values. Recently Chan and Chang designed such a scheme but their construction only allows a single secret to be shared per threshold value.

In this article we combine the previous two approaches to design a multiple time verifiable multi-secret sharing scheme where several secrets can be shared for each threshold value. Since the running time is an important factor for practical applications, we will provide a complexity comparison of our combined approach with respect to the previous schemes.

Keywords: Secret Sharing Scheme, Threshold Access Structures, Share Verifiability, Chinese Remainder Theorem, Keyed One-Way Functions.

1 Introduction

In 1979, Blakley and Shamir independently invented (t,n)-threshold secret sharing schemes in order to facilitate the distributed storage of secret data in an unreliable environment [1, 18]. Such a 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.

Sometimes, however, several secrets have to be shared simultaneously. A basic idea consists of using a (t,n)-threshold scheme as many times as the number of secrets.

Dingyi Pei et al. (Eds.): Inscrypt 2007, LNCS 4990, pp. 167–181, 2008.

This approach, however, is memory consuming. As noticed by Chien et al. [4], multi-secret sharing schemes can be used to overcome this drawback. In such a construction, multiple secrets are protected using the same amount of data usually needed to protect a single secret. Multi-secret sharing schemes can be classified into two families: one-time schemes and multiple time schemes [12]. One-time schemes imply the dealer must redistribute new shares to every participant once some particular secrets have been reconstructed. Such a redistribution process can be very costly both in time and resources, in particular, when the group size n gets large as it may be the case in group-oriented cryptography [6].

Several constructions of multiple time schemes have been achieved [4, 25]. Nevertheless they have the drawback that a dishonest dealer who distributes incorrect shares or a malicious participant who submits an invalid share to the combiner prevents the secrets from being reconstructed. The idea of robust computational secret sharing schemes was introduced by Krawczyk [14] to deal with this problem. Several such protocols were developed. Harn designed a verifiable multi-secret sharing scheme [10] which was extended by Lin and Wu [15]. In [3], Chang et al. recently improved that construction even further by providing resistance against cheating by malicious participants and reducing the computational complexity with respect to [10, 15]. The security of that scheme relies on the intractability of both factorization and discrete logarithm problem modulo a composite number. In [25], another multi-secret sharing scheme was developed by Yang et al. As [4], its security is based on the existence of keyed one-way functions introduced by Gong in [9]. Shao and Cao recently extended Yang et al.'s scheme by providing the verification property and reducing the number of public values[19].

It may occur that the same group of n participants share several secrets related to different threshold values according to their importance. As an example, consider that an army commander requests a strike to be executed and transmits the order to a group of 10 generals. One can imagine that any pair of officers can reconstruct the coordinates of the target and then initialize the process by mobilizing the appropriate equipment (plane, submarine, missile) but only subsets of 8 out of 10 generals can get access to the bomb activation code and launch the strike. Recently Chan and Chang designed such a construction [2] but it only allows a single secret to be shared per threshold value.

In this article, we propose a generalization of [2, 19] by introducing a Verifiable Multi-Threshold Multi-secret Sharing Scheme (VMTMSS) where several secrets can be shared per threshold value. The security of our multiple time scheme is guaranteed as soon as keyed one-way functions and collision resistant one-way functions exist. In the previous situation, our VMTMSS would enable any pair of generals to have access to target location, launch time, type of weapon to be used while any subset of 8 out of 10 officers can recover the bomb code as well as the commander's digital signature [20] as the approval for the strike. This example also emphasizes the need for computational efficiency. Therefore we will also provide an analysis of the computational cost of our construction.

This paper is organized as follows. In the next section we will recall the polynomial interpolation problem as well as Garner's algorithm since they will have an important role in our construction. In Sect. 3, we will describe our multi-secret sharing scheme and prove its soundness. In Sect. 4, we will analyze the computational complexity of our approach and compare it to the cost of the two constructions from [2, 19]. The last section will summarize the benefits of our construction.

2 Preliminaries

In this part we recall two problems which will play an important role in proving the soundness and efficiency of the scheme we describe in Sect. 3.

2.1 Interpolating Points

Assume that we are given λ points $(x_1, y_1), \ldots, (x_\lambda, y_\lambda)$ such that the x_i's are distinct in a field $\mathbb{K}$. The *Lagrange interpolating polynomial* $L_\lambda(X)$ is the only polynomial of degree at most $\lambda - 1$ passing through the previous λ points. Algorithm 4.6.1 from [8] computes the λ coefficients of $L_\lambda(X)$ using $\frac{5\,(\lambda-1)^2}{2}$ field operations in $\mathbb{K}$.

We now consider that we work over the finite field $\mathbb{Z}/p\mathbb{Z}$ for some prime number p. In this field an addition/subtraction requires $O(\log_2 p)$ bit operations and a multiplication needs $O(\log_2^2 p)$ bit operations. Using Algorithm 14.61 and Note 14.64 from [16], an inversion can be performed in $O(\log_2^2 p)$ bit operations as well. Therefore the λ coefficients of $L_\lambda(X)$ can be obtained using $O(\lambda^2\ \log_2^2 p)$ bit operations.

2.2 Solving the Chinese Remainder Problem

We first recall the Chinese Remainder Theorem (CRT):

Theorem 1. *Let $m_1, \ldots, m_\lambda$ be λ coprime integers and denote M their product. For any λ-tuple of integers $(v_1, \ldots, v_\lambda)$, there exists a unique x in $\mathbb{Z}/M\mathbb{Z}$ such that:*

$$\begin{cases} x \equiv v_1 \mod m_1 \\ \quad \vdots \qquad\quad \vdots \\ x \equiv v_\lambda \mod m_\lambda \end{cases}$$

Solving the Chinese remainder problem is reconstructing the unique x in $\mathbb{Z}/M\mathbb{Z}$ once $v_1, \ldots, v_\lambda$ and $m_1, \ldots, m_\lambda$ are given. This can be achieved thanks to Garner's algorithm [16]. Based on Note 14.74, its running time is $O(\lambda \log_2^2 M)$ bit operations.

3 Our Multi-secret Sharing Scheme

We assume that we have n participants $P_1, \ldots, P_n$ and ℓ distinct threshold values $t_1, \ldots, t_\ell$. Consider we have ℓ distinct prime numbers $p_1, \ldots, p_\ell$. For each i in $\{1, \ldots, \ell\}$ we denote $S_{i\,1}, \ldots, S_{i\,k_i}$ the k_i secrets of the (t_i, n)-threshold scheme. Without loss of generality we can assume that those k_i secrets belong to $\mathbb{Z}/p_i\mathbb{Z}$. We first introduce the following definition:

Definition 1. *A function $f : \mathbb{R}^+ \to \mathbb{R}^+$ is said to be* negligible *if:*

$$\forall \alpha > 0\, \exists \zeta_0 \in \mathbb{R}^+ \,:\, \forall \zeta > \zeta_0 \quad f(\zeta) < \zeta^{-\alpha}$$

We have the following definition adapted from Definition 13.2 [20].

Definition 2. *A threshold multi-secret sharing scheme for threshold value t is a method of sharing k secrets $S_1, \dots, S_k$ among a set of n participants $\{P_1, \dots, P_n\}$ in such a way that the following properties are satisfied:*
(i) (soundness) If at least t participants pool their shares together then they recover the whole k secrets $S_1, \dots, S_k$.
(ii) (secrecy) If at most $t-1$ participants pool their shares together then they do not recover the whole k secrets with non-negligible probability as a function of the secret's size.

The reader may notice that Definition 13.2 is related to perfect secrecy since it is there assumed that the coalition of $t-1$ participants does not know anything about the secret value (i.e. all values are equally probable). This cannot be held here as several secrets will be shared using the same polynomial. Nevertheless we will see that $t-1$ participants cannot recover the whole k secrets with good probability. We can generalize the previous definition as follows:

Definition 3. *A multiple-threshold multi-secret sharing scheme for threshold values $t_1, \dots, t_\ell$ is a method of sharing $k_1 + \cdots + k_\ell$ secrets $S_{1\,1}, \dots, S_{\ell\,k_\ell}$ among a set of n participants $\{P_1, \dots, P_n\}$ in such a way that the following properties are satisfied:*
(i) (soundness) For each $i \in \{1, \dots, \ell\}$, if at least t_i participants pool their shares together then they recover the whole k_i secrets $S_{i\,1}, \dots, S_{i\,k_i}$.
(ii) (secrecy) For each $i \in \{1, \dots, \ell\}$, if at most $t_i - 1$ participants pool their shares together then they do not recover the whole k_i secrets $S_{i\,1}, \dots, S_{i\,k_i}$ with non-negligible probability as a function of the secret's size.

A *verifiable multiple-threshold multi-secret sharing scheme* (VMTMSS) is a multiple-threshold multi-secret sharing scheme for which the validity of the share can be publicly verifiable. Let us introduce the following definition from [9]:

Definition 4. *A function $f(\cdot,\cdot)$ that maps a key and a second bit string of a fixed length is a* secure keyed one-way hash function *if it satisfies the following five properties:*

P1: *Given k and x, it is easy to compute $f(k,x)$.*
P2: *Given k and $f(k,x)$, it is hard to compute x.*
P3: *Without knowledge of k, it is hard to compute $f(k,x)$ for any x.*
P4: *Given k, it is hard to find two distinct values x and y such that $f(k,x) = f(k,y)$.*
P5: *Given (possibly many) pairs $(x, f(k,x))$, it is hard to compute k.*

Remark, however, this secure keyed one-way function is not equivalent to the two-variable one-way function defined by He and Dawson in [11] contrary to what claimed Chien et al. [4]. Indeed the collision resistance property P4 of the keyed one-way function is not a requirement for the functions created by He and Dawson (see Definition 1 in [11]).

We assume that we have ℓ such functions $f_1, \dots, f_\ell$ whose respective domains are $D_1, \dots, D_\ell$. Without loss of generality we can assume that the prime numbers $p_1, \dots, p_\ell$ are chosen such that: $\forall i \in \{1, \dots, \ell\}\ f_i(D_i) \subset \mathbb{Z}/p_i\mathbb{Z}$. We also assume: $\forall i \in \{1, \dots, \ell\}\ D_i \subset \mathbb{Z}/p_i\mathbb{Z} \times \mathbb{Z}/p_i\mathbb{Z}$. We need to use a collision resistant hash function H [17]. As in [13], it will be used to check the validity of the shares.

Our approach will consist of two steps. First we will treat each (t_i, n)-threshold scheme separately. We build a polynomial $F_i(X)$ whose degree and coefficients will be determined similarly to [25]. Second we will combine the ℓ polynomials $F_1(X), \ldots,$ $F_\ell(X)$ using the following result obtained by extending Corollary 3.2 from [2]:

Corollary 1. *(Polynomial form of CRT) Let $m_1, \ldots, m_\lambda$ be λ coprime integers and denote their product by M. For any λ-tuple of polynomials $(A_1(X), \ldots, A_\lambda(X))$ from $\mathbb{Z}/m_1\mathbb{Z}[X] \times \cdots \times \mathbb{Z}/m_\lambda\mathbb{Z}[X]$, there exists a unique polynomial $A(X)$ in $\mathbb{Z}/M\mathbb{Z}[X]$ such that:*

$$\begin{cases} A(X) \equiv A_1(X) \bmod m_1 \\ \quad\vdots \qquad\qquad \vdots \\ A(X) \equiv A_\lambda(X) \bmod m_\lambda \end{cases} \tag{1}$$

In addition: $\deg(A(X)) = \max_{i \in \{1,\ldots,\lambda\}} (\deg(A_i(X)))$.

Proof. In [2], Chan and Chang proved the existence of such a polynomial $A(X)$. What remains to demonstrate is its uniqueness and the value of its degree.

Let $A(X)$ be a polynomial from $\mathbb{Z}/M\mathbb{Z}[X]$ solution of (1) and denote α its degree. The ring isomorphism:

$$\mathbb{Z}/M\mathbb{Z} \simeq \mathbb{Z}/m_1\mathbb{Z} \times \cdots \times \mathbb{Z}/m_\lambda\mathbb{Z} \tag{2}$$

involves $\alpha = \max_{i \in \{1,\ldots,\lambda\}} (\deg(A_i(X)))$ since (2) implies an element μ is congruent to 0 in $\mathbb{Z}/M\mathbb{Z}$ if and only if μ is congruent to 0 in each $\mathbb{Z}/m_i\mathbb{Z}$ for $i \in \{1, \ldots, \lambda\}$.

Let $A(X)$ and $\tilde{A}(X)$ be two solutions of (1). Since their degree is α, we can write them as:

$$A(X) := \sum_{i=0}^{\alpha} a_i\, X^i \qquad \text{and} \qquad \tilde{A}(X) := \sum_{i=0}^{\alpha} \tilde{a}_i\, X^i$$

where the a_i's and $\tilde{a}_i$'s are elements of $\mathbb{Z}/M\mathbb{Z}$. Since these polynomials are solutions of (1) and due to (2), we deduce: $\forall i \in \{0, \ldots, \alpha\}\ a_i \equiv \tilde{a}_i \bmod M$. □

The previous proof involves that $A(X)$ can be computed from $A_1(X), \ldots, A_\lambda(X)$ using Garner's algorithm $\alpha + 1$ times. We will now present the details of our construction.

3.1 Scheme Construction

Our construction consists of three algorithms: SetUp, ShareConstruction and SecretReconstruction. The first two algorithms will be run by the dealer while the last one will be executed by the combiner. As in [4, 19], SetUp will only be run once while ShareConstruction will be called each time new secrets are to be shared. The private elements distributed to the n participants by the dealer when running SetUp will ensure that our VMTMSS is a multiple time scheme.

SetUp

Input: The group size n and ℓ distinct prime numbers $p_1, \ldots, p_\ell$.
1. For each $i \in \{1, \ldots, \ell\}$, generate n distinct elements of $\mathbb{Z}/p_i\mathbb{Z}$ denoted $s_{i\,1}, \ldots, s_{i\,n}$.
2. Use Garner's algorithm as: $\forall j \in \{1, \ldots, n\}\ \mathcal{S}_j := \text{Garner}(s_{1\,j}, \ldots, s_{\ell\,j}, p_1, \ldots, p_\ell)$.
3. Distribute $\mathcal{S}_j$ to participant P_j over a secure channel for each $j \in \{1, \ldots, n\}$.
Output: The n private values $\mathcal{S}_1, \ldots, \mathcal{S}_n$ which will be used by the participants to check the validity of their pseudo-shares.

We have the following observation concerning [4, 19]. Each of the n participants P_i receives a secret value s_i. The dealer chooses a random element r and evaluates the *pseudo-shares* $f(r, s_1), \ldots, f(r, s_n)$ where f is the keyed one-way function used in those schemes. He builds a polynomial $h(X)$ whose k lowest degree coefficients represent the k secrets to be shared. Finally he publishes $r, h(f(r, s_1)), \ldots, h(f(r, s_n))$ so that the combiner can verify the validity of shares. In order to ensure the multiple time property of their construction, a new value r is generated each time a new set of k secrets is to be shared. If r is chosen such that $f(r, s_{i_0})$ is 0 then P_{i_0} can recover one of the secrets as the constant term of the polynomial $h(X)$ from the list of public elements since: $h(0) = h(f(r, s_{i_0}))$. Even if the probability of such an event is negligible when the domain of f is large, it is still easy to deal with this problem by shifting each coefficient of the polynomial $h(X)$ by one position and setting up the new constant term as a random element. This is at the cost of publishing an extra point to reconstruct $h(X)$ since its degree has increased by 1.

We will now introduce our algorithm ShareConstruction. We first introduce the following notation:

$$\forall i \in \{1, \ldots, \ell\} \qquad \delta_i := \begin{cases} 0 & \text{if } t_i \geq k_i \\ k_i - t_i & \text{otherwise} \end{cases}$$

Notice that δ_i can be computed as soon as both t_i and k_i are known.

ShareConstruction

Input: The group size n, the prime numbers $p_1, \ldots, p_\ell$, the threshold values $t_1, \ldots, t_\ell$, the number of secrets $k_1, \ldots, k_\ell$, the corresponding secrets $S_{1\,1}, \ldots, S_{1\,k_1}, \ldots, S_{\ell\,1}, \ldots, S_{\ell\,k_\ell}$, the functions $f_1, \ldots, f_\ell$, the elements $s_{1\,1}, \ldots, s_{\ell\,n}$ from SetUp and the collision resistant hash function H.

1. For each $i \in \{1, \ldots, \ell\}$, pick uniformly at random an element r_i from $\mathbb{Z}/p_i\mathbb{Z}$. Use Garner's algorithm as: $\mathcal{R} := \text{Garner}(r_1, \ldots, r_\ell, p_1, \ldots, p_\ell)$.
2. Do the following:
   2.1. Compute $f_i(r_i, s_{i\,j})$ for $i \in \{1, \ldots, \ell\}$ and $j \in \{1, \ldots, n\}$.
   2.2. Compute the hashes $H(f_i(r_i, s_{i\,j}))$ for $i \in \{1, \ldots, \ell\}$ and $j \in \{1, \ldots, n\}$ and publish them as table T_{H}.
   2.3. Use Garner's algorithm as: $\forall j \in \{1, \ldots, n\}\ \mathcal{P}_j := \text{Garner}(f_1(r_1, s_{1\,j}), \ldots, f_\ell(r_\ell, s_{\ell\,j}), p_1, \ldots, p_\ell)$.
3. For each $i \in \{1, \ldots, \ell\}$ do the following:
   3.1. Pick uniformly at random an element C_i from $\mathbb{Z}/p_i\mathbb{Z}$.
   3.2. If $t_i > k_i$ then:

Pick uniformly at random $u_{i\,1}, \ldots, u_{i\,\delta_i}$ from $\mathbb{Z}/p_i\mathbb{Z}$.

Build the polynomial: $F_i(X) := C_i + \sum_{j=1}^{k_i} S_{i\,j}\, X^j + \sum_{j=1}^{t_i-k_i} u_{i\,j}\, X^{j+k_i}$

Else

Build the polynomial: $F_i(X) := C_i + \sum_{j=1}^{k_i} S_{i\,j}\, X^j$

4. Denote $D := \max_{i\in\{1,\ldots,\ell\}}(\deg(F_i(X)))$. For each $i \in \{1,\ldots,\ell\}$, write $F_i(X)$ as: $F_i(X) := \sum_{j=0}^{D} F_{i\,j}\, X^j$ where: $\forall j \in \{\deg(F_i(X))+1,\ldots,D\}$ $F_{i\,j} = 0$. Use Garner's algorithm as: $\forall j \in \{0,\ldots,D\}$ $\mathcal{F}_j :=$ Garner$(F_{1\,j},\ldots,F_{\ell\,j},p_1,\ldots,p_\ell)$.
5. Build the polynomial $\mathcal{F}(X)$ as: $\mathcal{F}(X) := \sum_{j=0}^{D} \mathcal{F}_j\, X^j$ and compute $\mathcal{F}(\mathcal{P}_1),\ldots,\mathcal{F}(\mathcal{P}_n)$.
6. Do the following:
   6.1. For each $i \in \{1,\ldots,\ell\}$, generate an element a_i from $\mathbb{Z}/p_i\mathbb{Z}$ distinct from $s_{i\,1},\ldots,s_{i\,n}$.
   6.2. Use Garner's algorithm as: $\mathcal{A} :=$ Garner$(f_1(r_1,a_1),\ldots,f_\ell(r_\ell,a_\ell),p_1,\ldots,p_\ell)$.
   6.3. Compute $\mathcal{F}(\mathcal{A})$.
7. For each $i \in \{1,\ldots,\ell\}$ such that $\delta_i > 0$ do the following:
   7.1. Generate δ_i elements $s'_{i\,1},\ldots,s'_{i\,\delta_i}$ such that $s_{i\,1},\ldots,s_{i\,n},a_i,s'_{i\,1},\ldots,s'_{i\,\delta_i}$ are $n+1+\delta_i$ distinct elements of $\mathbb{Z}/p_i\mathbb{Z}$.
   7.2. Compute $f_i(r_i,s'_{i\,1}),\ldots,f_i(r_i,s'_{i\,\delta_i})$.
   7.3. Compute $F_i(f_i(r_i,s'_{i\,1})),\ldots,F_i(f_i(r_i,s'_{i\,\delta_i}))$.
8. Publish the table T containing $\mathcal{R},\mathcal{F}(\mathcal{P}_1),\ldots,\mathcal{F}(\mathcal{P}_n),(\mathcal{A},\mathcal{F}(\mathcal{A}))$ as well as the couples $(f_i(r_i,s'_{i\,1}),F_i(f_i(r_i,s'_{i\,1}))),\ldots,(f_i(r_i,s'_{i\,\delta_i}),F_i(f_i(r_i,s'_{i\,\delta_i})))$ for each i such that $\delta_i > 0$.

Output: The table T_{H} which will be used to verify the pseudo-shares and the table T which will be used to reconstruct the secrets of our VMTMSS.

Notice that $(\mathcal{A},\mathcal{F}(\mathcal{A}))$ is the extra point needed to overcome the problem from [19]. We also remark that any participant P_j can compute the pseudo-shares $f_i(r_i,s_{i\,j})$ from the public value R and his secret element $\mathcal{S}_j$ since:

$$\begin{cases} r_i = \mathcal{R} \mod p_i \\ s_{i\,j} = \mathcal{S}_j \mod p_i \end{cases}$$

Using this information any participant can verify the validity of his pseudo-shares by checking their ℓ hashes from table T_{H}. Similarly the combiner can check the validity of any pseudo-share submitted during the secret reconstruction process using T_{H} as well. Notice, however, that the prime numbers $p_1,\ldots,p_\ell$ should be large enough in order to prevent an exhaustive search to be performed by an adversary who would compute $H(\zeta)$ (where $\zeta \in \mathbb{Z}/p_i\mathbb{Z}$) until finding a match amongst the n elements $H(f_i(r_i,s_{i\,1})),\ldots,H(f_i(r_i,s_{i\,n}))$. Figure 1 represents the previous two algorithms. The construction of polynomials $F_1(X),\ldots,F_\ell(X)$ and $\mathcal{F}(X)$ is depicted on Fig. 2.

We will now design SecretReconstruction which is run be combiner to recover the secrets. We assume that $P_{j_1},\ldots,P_{j_{t_i}}$ are the t_i participants wishing to reconstruct the k_i secrets of the (t_i,n)-threshold scheme.

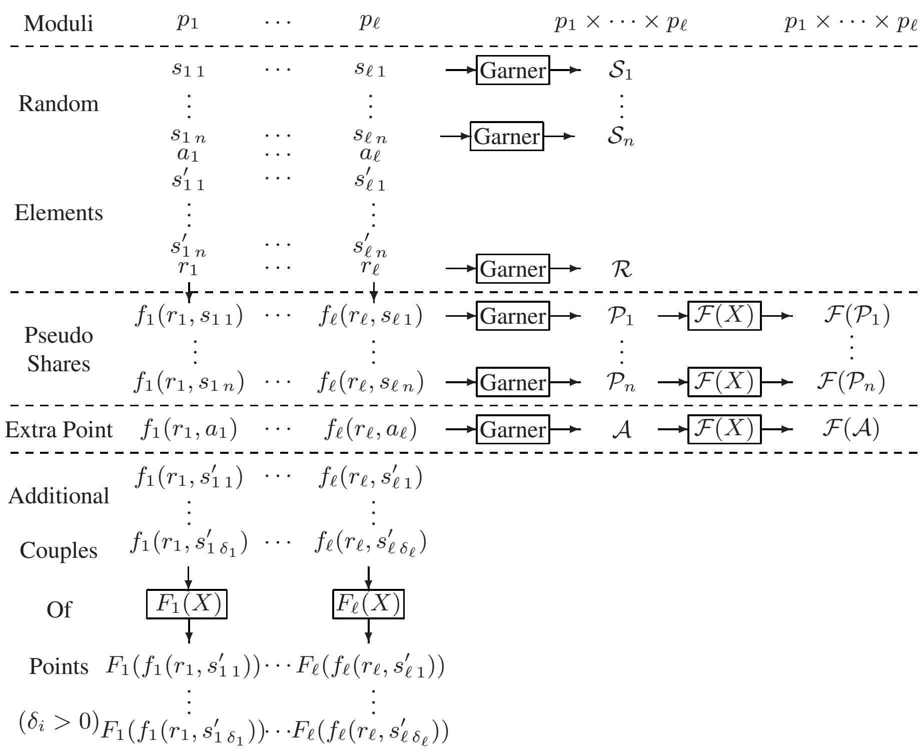

Fig. 1. Representation of SetUp and ShareConstruction

SecretReconstruction

Input: The threshold value t_i, the number of secrets k_i, the prime numbers $p_1, \ldots, p_\ell$, the public table T as well as the pseudo-shares of the t_i participants $f_i(r_i, s_{i\,j_1}), \ldots, f_i(r_i, s_{i\,j_{t_i}})$.

1. Compute $x_{t_i+1} := \mathcal{A} \bmod p_i$ and $y_{t_i+1} := \mathcal{F}(\mathcal{A}) \bmod p_i$. For each $\lambda \in \{1, \ldots, t_i\}$, compute $y_\lambda := \mathcal{F}(\mathcal{P}_{j_\lambda}) \bmod p_i$.

2. If $\delta_i = 0$ then:

 2.1. Reconstruct the Lagrange interpolating polynomial passing through the points $(f_i(r_i, s_{i\,j_1}), y_1), \ldots, (f_i(r_i, s_{i\,j_{t_i}}), y_{t_i}), (x_{t_i+1}, y_{t_i+1})$.

 2.2. Write the polynomial obtained as: $\sum_{j=0}^{t_i} \mu_j\, X^j$ and return $\mu_1, \ldots, \mu_{k_i}$.

 Else

 2.3. Reconstruct the Lagrange interpolating polynomial passing through the points $(f_i(r_i, s_{i\,j_1}), y_1), \ldots, (f_i(r_i, s_{i\,j_{t_i}}), y_{t_i}), (x_{t_i+1}, y_{t_i+1}), (f_i(r_i, s'_{i\,1}), F_i(f_i(r_i, s'_{i\,1}))), \ldots, (f_i(r_i, s'_{i\,\delta_i}), F_i(f_i(r_i, s'_{i\,\delta_i})))$.

 2.4. Write the polynomial obtained as: $\sum_{j=0}^{k_i} \mu_j\, X^j$ and return $\mu_1, \ldots, \mu_{k_i}$.

Output: The k_i secrets $\mu_1, \ldots, \mu_{k_i}$ of the (t_i, n)-threshold scheme.

$$\begin{array}{lcccccccl}
\text{Moduli} & & & & & & & & \\
p_1 & S_{1\,1} & \cdots & S_{1\,k_1} & F_{1\,0} & \cdots & F_{1\,\max(t_1,k_1)} & \} & F_1(X) \\
\vdots & \vdots & & \vdots & \vdots & & \vdots & & \vdots \\
p_\ell & S_{\ell\,1} & \cdots & S_{\ell\,k_\ell} & F_{\ell\,0} & \cdots & F_{\ell\,\max(t_\ell,k_\ell)} & \} & F_\ell(X)
\end{array}$$

$$\begin{array}{lccc}
p_1 & F_{1\,0} & \cdots & F_{1\,D} \\
\vdots & \vdots & & \vdots \\
p_\ell & F_{\ell\,0} & \cdots & F_{\ell\,D} \\
 & \text{Garner} & & \text{Garner} \\
p_1 \times \cdots \times p_\ell & \multicolumn{3}{c}{\underbrace{\mathcal{F}_0 \quad \cdots \quad \mathcal{F}_D}_{\mathcal{F}(X)}}
\end{array}$$

Fig. 2. Construction of Polynomials by the Dealer

3.2 Security Analysis

In this section, we have to demonstrate that our VMTMSS verifies the properties from Definition 3. In particular we have to argue that the table of hashes T_{H} and the table of extra points T do not leak too much information about the secrets. We have the following result for our construction:

Theorem 2. *The reconstruction algorithm* SecretReconstruction *is sound.*

Proof. We have to demonstrate that, for any value i in $\{1, \ldots, \ell\}$, the elements output by SecretReconstruction are the k_i secrets of the (t_i, n)-threshold scheme whatever the family of t_i participants is.

Let i be any element of $\{1, \ldots, \ell\}$. Consider $P_{j_1}, \ldots, P_{j_{t_i}}$ a family of t_i participants. Due to Steps $2, 4$ and 5 of ShareConstruction2, we have the following result:

$$\forall i \in \{1, \ldots, \ell\}\, \forall \lambda \in \{1, \ldots, t_i\}\; F_i(f_i(r_i, s_{i\,j_\lambda})) = \mathcal{F}(\mathcal{P}_{j_\lambda}) \bmod p_i$$

Due to Property P4 of f_i, Step 1 of SetUp and Step 6.1 of ShareConstruction, the elements $f_i(r_i, s_{i\,j_1}), \ldots, f_i(r_i, s_{i\,j_{t_i}}), f_i(r_i, a_i)$ are distinct with overwhelming probability. Since $f_i(r_i, a_i) = \mathcal{A} \bmod p_i = x_{t_i+1}$, the $t_i + 1$ points $(f_i(r_i, s_{i\,j_1}), y_1), \ldots,$ $(f_i(r_i, s_{i\,j_{t_i}}), y_{t_i}), (x_{t_i+1}, y_{t_i+1})$ have different abscissas in $\mathbb{Z}/p_i\mathbb{Z}$. We have two cases to consider:

First Case: $\delta_i = 0$**.** We can interpolate the previous $t_i + 1$ points as in Sect. 2.1 and denote $L_{t_i+1}(X)$ the corresponding Lagrange polynomial obtained at Step 2.1 of SecretReconstruction. It should be noticed that the polynomial $F_i(X)$ defined at Step 3.2

of ShareConstruction passes through the same points and has degree at most t_i (it is exactly t_i if the highest degree coefficient is different from 0). Due to the uniqueness of such a polynomial (see Sect. 2.1) we get: $L_{t_i+1}(X) = F_i(X)$. Thus the k_i coefficients returned at Step 2.2 of SecretReconstruction are the k_i secrets of the (t_i, n)-threshold scheme: $S_{i\,1}, \ldots, S_{i\,k_i}$.

Second Case: $\delta_i > 0$**.** Using table T, we obtain δ_i additional points: $(f_i(r_i, s'_{i\,1}), F_i(f_i(r_i, s'_{i\,1}))), \ldots, (f_i(r_i, s'_{i\,\delta_i}), F_i(f_i(r_i, s'_{i\,\delta_i})))$. This leads to a total of $t_i + 1 + \delta_i = k_i + 1$ points have different abscissas. We can interpolate those $k_i + 1$ points as in Sect. 2.1 and denote $L_{k_i+1}(X)$ the corresponding Lagrange polynomial obtained at Step 2.3 of SecretReconstruction. As $F_i(X)$ passes through the same points and has degree at most k_i (it is exactly k_i if the secret $S_{i\,k_i}$ is different from 0) we get: $L_{k_i+1}(X) = F_i(X)$. Thus the k_i coefficients returned at Step 2.4 of SecretReconstruction are the k_i secrets of the (t_i, n)-threshold scheme: $S_{i\,1}, \ldots, S_{i\,k_i}$. □

Theorem 3. *Our VMTMSS achieves secrecy.*

Proof. Let i be any integer in $\{1, \ldots, \ell\}$. Assume that $t_i - 1$ participants pool their pseudo-shares together and use public knowledge from tables T and T_{H}. The participants are denoted $P_{j_1}, \ldots, P_{j_{t_i-1}}$. Since H is a collision resistant hash function, H is a one-way function. Therefore with overwhelming probability, the only information the colluders learn from table T_{H} is the pseudo-shares of the non-colluding members are different from theirs. Nevertheless this fact was already known to each of the n participants due to Step 1 of SetUp, property P4 and (2). So table T_{H} does not give any extra-information to the colluders with overwhelming probability. We have two cases to consider.

First Case: $\delta_i = 0$**.** The colluders have to determine the $t_i + 1$ coefficients of $F_i(X)$ (Step 3.2 of ShareConstruction). Using the same technique as in the proof of Theorem 2, they can obtain t_i points $F_i(X)$ goes through from their pseudo-shares and the point $(\mathcal{A}, \mathcal{F}(\mathcal{A}))$ from T. Consider the set:

$$E := \{(f_i(r_i, s_{i\,j}), F_i(f_i(r_i, s_{i\,j}))) : j \notin \{j_1, \ldots, j_{t_i-1}\}\}$$

The elements of E represent the points owned by the non-colluding members. It should be noticed that the n values $F_i(f_i(r_i, s_{i\,1})), \ldots, F_i(f_i(r_i, s_{i\,n}))$ are known to each group participant since they can be obtained by reductions modulo p_i from elements $\mathcal{F}(\mathcal{P}_1), \ldots, \mathcal{F}(\mathcal{P}_n)$ contained in T. We will see that the probability the colluders can construct an element of E is negligible as a function of the length of p_i.

Due to Property P4 of the function f_i the colluders know, with overwhelming probability, that the abscissas of the elements of E belong to:

$$f_i(D_i) \setminus \left\{f_i(r_i, s_{i\,j_1}), \ldots, f_i(r_i, s_{i\,j_{t_i-1}}), \mathcal{A} \bmod p_i\right\}$$

We would like to draw the reader's attention to the following point. Once $F_i(f_i(r_i, s_{i\,\mu}))$ is given, there may be more than one value x such that $F_i(x) = F_i(f_i(r_i, s_{i\,\mu}))$. In the worst case we can have up to $n - t_i + 1$ such values for x which happens when all the ordinates of the elements of E are equal. Thus:

$$\text{Prob}((x, F_i(f_i(r_i, s_{i\,\mu}))) \in E, x \text{ is built by the colluders}) \leq \frac{n+1}{|f_i(D_i)| - n}$$

Second Case: $\delta_i > 0$**.** The colluders have to determine the $k_i + 1$ coefficients of $F_i(X)$ (Step 3.2 of ShareConstruction). As before, they can obtain $t_i + \delta_i$ points $F_i(X)$ goes through from their pseudo-shares and the $\delta_i + 1$ points from T. As previously we get:

$$\text{Prob}((x, F_i(f_i(r_i, s_{i\,\mu}))) \in E, x \text{ is built by the colluders}) \leq \frac{n+1}{|f_i(D_i)| - k_i}$$

Without loss of generality, we can assume that the range of f_i represents a non-negligible part of $\mathbb{Z}/p_i\mathbb{Z}$. At the same time, we can consider that the group size n and k_i is small in comparison to p_i so that there exists C_i, independent from p_i, such that, in both cases, we have:

$$\text{Prob}((x, F_i(f_i(r_i, s_{i\,\mu}))) \in E, x \text{ is built by the colluders}) \leq \frac{C_i}{p_i}$$

Therefore it is sufficient to pick the smallest of the ℓ primes to be 80 bits long to ensure computational secrecy for our scheme. $\square$

4 Complexity Survey

As claimed in Sect. 1, the computational and storage costs represent key factors to take into account when implementing a protocol as a part of a commercial application. In this part we study the cost of our construction and compare it to the schemes from [2, 19]. In this section we denote M the product of the ℓ prime numbers $p_1, \ldots, p_\ell$. We assume that picking random elements from the sets $\mathbb{Z}/p_1\mathbb{Z}, \ldots, \mathbb{Z}/p_\ell\mathbb{Z}$ has a negligible computational cost.

4.1 Cost of Our Construction

Computational Cost at the Dealer. Based on Sect. 2.2, SetUp can be executed in $O(n\,\ell \log_2^2 M)$ bit operations.

ShareConstruction performs $n + D + 3$ calls to Garner's algorithm which results in $O((n+D)\,\ell\,\log_2^2 M)$ bit operations. In addition there are $n+1$ polynomial evaluations over $\mathbb{Z}/M\mathbb{Z}$. Using Horner's rule each of them can be done via D additions and D multiplications in $\mathbb{Z}/M\mathbb{Z}$. Based on Sect. 2.1, this represents a total of $O(n\,D\,\log_2^2 M)$ bit operations. There are also δ_i polynomial evaluations over $\mathbb{Z}/p_i\mathbb{Z}$. If we denote $\Delta := \max_{i\in\{1,\ldots,\ell\}} \delta_i$ then the $\delta_1 + \cdots + \delta_\ell$ polynomial evaluations cost $O\left(\Delta\, D \log_2^2 \left(\max_{i\in\{1,\ldots,\ell\}} p_i\right)\right)$ bit operations. Since each prime number p_i is less than

M, the total cost of ShareConstruction is $O([D\,(\ell+n+\Delta)+n\,\ell]\,\log_2^2 M)$ bit operations. Furthermore the collision resistant hash function H is run $n\,\ell$ times while each keyed one-way function f_i is run $n+\delta_i$ times.

Computational Cost at the Combiner. Notice that the cost of SecretReconstruction depends on the threshold value t_i. We have t_i+2 reductions modulo p_i of elements $\mathbb{Z}/M\mathbb{Z}$. This can be done using Euclid's divisions in $O(t_i\,(\log_2 M\ \log p_i))$ bit operations. In addition an interpolating polynomial passing through $t_i+1+\delta_i$ points is to be build over $\mathbb{Z}/p_i\mathbb{Z}$. We know from Sect. 2.1 this can be achieved in $O((t_i+\delta_i)^2\ \log_2^2 p_i)$ bit operation. Since $p_i \leq M$, we deduce that SecretReconstruction runs in $O((t_i+\delta_i)^2\ \log_2 M\ \log_2 p_i)$ bit operations.

Storage of Public Elements. Denote $\mathrm{size}(x)$ the number of bits used to represent the natural integer x. We have $\mathrm{size}(x) = \lfloor \log_2 x \rfloor + 1$. We define $\rho := \sum_{i=1}^{\ell} \delta_i\,\mathrm{size}(p_i)$ and $\rho' := \sum_{i=1}^{\ell} \mathrm{size}(p_i)$. We also denote $\mathcal{H}$ the bitsize of a digest produced by the collision resistant hash function. First, storing T_{H} requires $n\,\ell\,\mathcal{H}$ bits. Second, T contains $n+3$ elements from $\mathbb{Z}/M\mathbb{Z}$ and $2\,\delta_i$ elements from $\mathbb{Z}/p_i\mathbb{Z}$ for each $i \in \{1,\ldots,\ell\}$. Thus the size of T is $(n+3)\,\mathrm{size}(M)+2\,\rho$ bits. As a consequence the size of public elements represents a total of $n\,(\ell\,\mathcal{H}+\mathrm{size}(M))+3\,\mathrm{size}(M)+2\,\rho$ bits. Notice, however, that the sender must buffer all the elements $s_{1\,1},\ldots,s_{\ell\,n}$ from Step 1 of SetUp which represents $n\,\rho'$ bits.

4.2 Efficiency Comparison

The parameters of the schemes are depicted in Table 1. Notice that the construction by Chan and Chang does not allow flexibility in the number of secrets to be shared. Indeed when we iterate that construction λ times (with the same threshold values) then the total number of secrets has to be $\lambda\,\ell$. Therefore we restrict our comparison to the scheme by Shao and Cao as it enables to choose the number of secrets per threshold independently from the total number of thresholds. Remark that our construction can be seen as extension of Chan and Chang's approach providing flexibility. To have an accurate survey, we assume that Shao and Cao's construction is iterated ℓ times (one iteration per family of k_i secrets). The results of our comparison are summarized in Table 2.

The reader can notice that ρ' is slightly larger than $\mathrm{size}(M)$ so, a priori, our technique does not provide any significant size benefit from ℓ iterations of Shao and Cao's construction. As noticed in [2], however, the latter approach requires each participant to keep multiple shares which can create a share management problem. In our construc-

Table 1. Parameters of the Three VMTMSS

	Our Scheme	Chan-Chang's Scheme [2]	Shao-Cao's Scheme [19]
Thresholds	ℓ	ℓ	1
Secrets per Threshold	k_i	1	k
Size Private Values	$\mathrm{size}(M)$ bits	$\mathrm{size}(p)$ bits	$\mathrm{size}(p)$ bits

Table 2. Computational Complexity of the Three VMTMSS

	Our Scheme	Shao-Cao's Scheme [19]
Size Private Values	size(M) bits	ρ' bits
Set-up Phase	$n\,\ell$ random elements n calls to Garner	$n\,\ell$ random elements
Share Creation Process	δ_i pol. eval. in each $\mathbb{Z}/p_i\mathbb{Z}$ $n+1$ pol. eval. in $\mathbb{Z}/M\mathbb{Z}$ $n\,\ell$ calls to H $n+\delta_i$ calls to each f_i $n+D+3$ calls to Garner	$n+\delta_i$ pol. eval. in each $\mathbb{Z}/p_i\mathbb{Z}$ $\max(t_i,k_i)$ exp. in each $\mathbb{Z}/p_i\mathbb{Z}$ n calls to each f_i
Pseudo-Share Validity Verification	1 call to H	$\max(t_i,k_i)$ exp. in each $\mathbb{Z}/p_i\mathbb{Z}$ $\max(t_i,k_i)$ exp. in $\mathbb{Z}/\frac{p_i-1}{2}\mathbb{Z}$ $\max(t_i,k_i)$ mult. in $\mathbb{Z}/p_i\mathbb{Z}$
Secret Recovery	1 polynomial reconstruction t_i+2 reductions modulo p_i	1 polynomial reconstruction
Storage Public Elements	$n\,(\ell\,\mathcal{H}+\text{size}(M))+3\,\text{size}(M)+2\,\rho$ bits	$(n+1)\,\rho'+2\,\rho+\sum_{i=1}^{\ell} t_i\,\text{size}(p_i)$ bits
Storage Sender	$n\,\rho'$ bits	$n\,\rho'$ bits

tion, each participant holds a single "master" share which can be used to recreate the share for each (t_i,n)-scheme. We now have two points to consider.

First, the pseudo-share verification process from [19] is expensive. Indeed verifying a single pseudo-share roughly costs $2\,\max(t_i,k_i)$ exponentiations in $\mathbb{Z}/p_i\mathbb{Z}$. Even if each of them can be performed in $O(\log_2^3 p_i)$ bit operations using the fast exponentiation. algorithm [17], the coefficient $\max(t_i,k_i)$ is prohibitive for large thresholds t_i. In addition, when the communication channel is under attack of malicious users flooding the combiner with incorrect values, the coefficient $\max(t_i,k_i)$ may result in successful denial-of-service attacks as the computational resources needed to identify correct shares amongst forgeries become too large. This problem does not happen with our construction as only a single hash as to be computed to validate/discard a share. Notice that each participant first needs to perform 2 reductions modulo p_i and 1 call to f_i to construct his pseudo-share from his secret value and the public element $\mathcal{R}$. However this is at the cost of running $2\,n+D+3$ times Garner's algorithm at the dealer during the set-up and share construction phases.

Second, our pseudo-share verification process requires $n\ell$ hashes to be published as table T_{H}. If we use SHA-256 as collision resistant hash function then T_{H} is represented over $256\,n\,\ell$ bits. On the other hand, the construction by Shao and Cao is secure provided that the discrete logarithm problem over each $\mathbb{Z}/p_i\mathbb{Z}$ is intractable. For achieve security, it is suggested to use 1024-bit moduli or larger [16]. If we assume that the different thresholds are roughly equal to the same value t then the coefficient

$\sum_{i=1}^{\ell} t_i \, \mathrm{size}(p_i)$ is approximately $1024\,\ell\,t$ bits. Therefore the storage of our public elements less expensive as soon as $t \geq \frac{n}{4}$, i.e. the construction by Shao and Cao provides better space efficiency only for small threshold values.

5 Conclusion

In this paper, we generalized the approaches from [2, 19] by designing a multiple time verifiable secret sharing scheme allowing several secrets to be shared per threshold value. As in [19], our construction allows any number of secrets to be shared per threshold value. In addition, we showed that our pseudo-share verification process was much faster than in [19] while the storage requirements were smaller. We would like to point three facts. First, we assumed that the threshold values were different (see Sect. 3). Nevertheless our techniques could also be employed if some threshold t_i is used τ_i times provided that different primes $p_{i\,1}, \ldots, p_{i\,\tau_i}$ are used respectively. Second, the security of our scheme is based on the random oracle model for the collision resistant hash function H. Most hash functions used in practice are considered heuristically collision resistant. Recently several such functions were successfully attacked [21, 22, 23, 24, 26]. In order to maintain the security of our protocol, we suggest to use a hash function whose security has be proved to be linked to a computationally difficult problem such as Very Smooth Hash [5] or Gibson's discrete logarithm-based hash function [7]. Nevertheless this may result into larger digests or increased running time. Finally the main drawback of our construction is that we are only able to deal with threshold schemes and our approaches cannot be directly generalized to non-threshold access structures.

Acknowledgement

The authors are 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, DP0665035 and DP0663452. This work was supported in part by the National Natural Science Foundation of China Grant 60553001 and the National Basic Research Program of China Grant 2007CB807900, 2007CB807901. Christophe Tartary did most of this work while at Macquarie University where his research was supported by an iMURS scholarship. The research of Huaxiong Wang is partially supported by the Minitry of Education of Singapore under grant T206B2204.

References

[1] Blakley, G.R.: Safeguarding cryptographic keys. In: AFIPS 1979 National Computer Conference, pp. 313–317. AFIPS Press (1979)

[2] Chan, C.-W., Chang, C.-C.: A scheme for threshold multi-secret sharing. Applied Mathematics and Computation 166(1), 1–14 (2005)

[3] Chang, T.-Y., Hwang, M.-S., Yang, W.-P.: An improvement on the Lin-Wu (t, n) threshold verifiable multi-secret sharing scheme. Applied Mathematics and Computation 163(1), 169 (2005)

[4] Chien, H.-Y., Jan, J.-K., Tseng, Y.-M.: A practical (t, n) multi-secret sharing. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences E83-A(12), 2762–2765 (2000)
[5] Contini, S., Lenstra, A.K., Steinfeld, R.: VSH: an efficient and provable collision resistant hash collision. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 165–182. Springer, Heidelberg (2006)
[6] Desmedt, Y.: Society and group oriented cryptography: A new concept. In: Pomerance, C. (ed.) CRYPTO 1987. LNCS, vol. 293, pp. 120–127. Springer, Heidelberg (1988)
[7] Gibson, J.K.: Discrete logarithm hash function that is collision free and one way. IEE Proceedings - Computers and Digital Techniques 138(6), 407–410 (1991)
[8] Golub, G.H., Van Loan, C.F.: Matrix Computations, 3rd edn. The Johns Hopkins University Press (1996)
[9] Gong, L.: New protocols for third-party-based authentication and secure broadcast. In: 2^{nd} ACM Conference on Computer and Communications Security, pp. 176–183. ACM Press, New York (1994)
[10] Harn, L.: Efficient sharing (broadcast) of multiple secrets. IEE Proceedings - Computers and Digital Techniques 142(3), 237–240 (1995)
[11] He, J., Dawson, E.: Multisecret sharing scheme based one-way function. IEE Electronic Letters 31(2), 93–95 (1995)
[12] Jackson, W.-A., Martin, K.M., O'Keefe, C.M.: On sharing many secrets (extended abstract). In: Safavi-Naini, R., Pieprzyk, J.P. (eds.) ASIACRYPT 1994. LNCS, vol. 917, pp. 42–54. Springer, Heidelberg (1995)
[13] Karnin, E.D., Greene, J.W., Hellman, M.E.: On secret sharing systems. IEEE Transactions on Information Theory 29(1), 35–41 (1983)
[14] Krawczyk, H.: Secret sharing made short. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 136–146. Springer, Heidelberg (1994)
[15] Lin, T.-Y., Wu, T.-C.: (t, n) threshold verifiable multisecret sharing scheme based on factorisation intractability and discrete logarithm modulo a composite problems. IEE Proceedings - Computers and Digital Techniques 146(5), 264–268 (1999)
[16] Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1996)
[17] Pieprzyk, J., Hardjono, T., Seberry, J.: Fundamentals of Computer Security. Springer, Heidelberg (2003)
[18] Shamir, A.: How to share a secret. Communication of the ACM 22(11), 612–613 (1979)
[19] Shao, J., Cao, Z.: A new efficient (t, n) verifiable multi-secret sharing (VMSS) based on YCH scheme. Applied Mathematics and Computation 168(1), 135–140 (2005)
[20] Stinson, D.R.: Cryptography: Theory and Practice, 3rd edn. Chapman & Hall/CRC (2006)
[21] Wang, X., Lai, X., Feng, D., Chen, H., Yu, X.: Cryptanalysis of the hash functions MD4 and RIPEMD. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 1–18. Springer, Heidelberg (2005)
[22] Wang, X., Yin, Y.L., Yu, H.: Finding collisions in the full SHA-1. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 17–36. Springer, Heidelberg (2005)
[23] Wang, X., Yu, H.: How to break MD5 and other hash functions. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 19–35. Springer, Heidelberg (2005)
[24] Wang, X., Yu, H., Yin, Y.L.: Efficient collision search attacks on SHA-0. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 1–16. Springer, Heidelberg (2005)
[25] Yang, C.-C., Chang, T.-Y., Hwang, M.-S.: A (t, n) multi-secret sharing scheme. Applied Mathematics and Computation 151(2), 483–490 (2004)
[26] Yu, H., Wang, G., Zhang, G., Wang, X.: The second-preimage attack on MD4. In: Desmedt, Y.G., Wang, H., Mu, Y., Li, Y. (eds.) CANS 2005. LNCS, vol. 3810, pp. 1–12. Springer, Heidelberg (2005)