Space Efficient and Integrity-ensured Secret Sharing Schemes for Multimedia Secrecy without Recursion

Wei-Liang Tai[1], Ya-Fen Chang[2]

[1] Department of Information Communications, Chinese Culture University, Taiwan
[2] Department of Computer Science and Information Engineering,
National Taichung University of Science and Technology, Taiwan
dwl@ulive.pccu.edu.tw, cyf@nutc.edu.tw

Abstract

Multimedia is content that uses a combination of *different* content forms. To share multimedia among multiple users secretly becomes a tough challenge because the amount of transmitted or involved data might be large. To ensure efficiency and security for multimedia secret sharing, we propose two secret sharing schemes without recursion in this paper, where the second scheme can control the security of produced shares. The proposed schemes ensure both security and efficiency and further provide a mechanism to check the integrity of the recovered secret.

Keywords: Multimedia, Secret sharing, Recursion, Integrity

1 Introduction

Multimedia is content that uses a combination of *different* content forms such as text, images, audio, video and interactive content. Due to its variety, multimedia might be recorded, played, displayed, or interacted with by some processing devices. To increase the security level of multimedia information, the multimedia information needs to be shared among multiple users, and only a certain number of them can cooperate to retrieve the original multimedia information. Unlike text, the amount of transmitted or involved multimedia information might be large. This makes share multimedia among multiple users secretly becomes a tough challenge [1].

Shamir first proposed the concept of secret sharing in 1979 [2]. Shamir's secret sharing scheme possesses a distinguishing property such that the secret can be shared among n users and reconstructed by only k users. It is well-known to be k-out-of-n secret sharing. However, the size of n shares is the same as or greater than that of the secret. This makes Shamir's secret sharing scheme lack space efficiency and not suitable for multimedia information.

In 2011, Parakh and Kak [3] proposed a secret sharing scheme for implicit data security by modifying Krawczyk's scheme [4]. In Parakh and Kak's secret sharing scheme, the size of shares is only $1/(k\text{-}1)$ of that in Shamir's scheme, and a recursion algorithm is adopted to generate corresponding polynomials. Parakh and Kak claimed that their scheme possessed superior space efficiency. Meanwhile, they also proposed a mechanism to control the security of produced shares. However, Tai et al. found that both of Parakh and Kak's secret sharing schemes suffer from degree degeneration and gave counter examples [5]. This flaw results from that the coefficient of x^{k-1} is automatically generated by a recursive algorithm. That is, the coefficient of x^{k-1} might be 0 such that the shared secret can be retrieved when the number of involved shareholders is less than the legality threshold k. Although this flaw makes Parakh and Kak's secret sharing schemes violate the concept of secret sharing, the technique to reduce the size of shares can be adopted to ensure space efficiency. On the other hand, a recursion algorithm is adopted in Parakh and Kak's secret sharing schemes to generate corresponding polynomials. This approach makes Parakh and Kak's schemes lack computational efficiency. How to preserve space efficiency while enhancing computational efficiency of Parakh and Kak's schemes is the key to designing a secret sharing scheme for multimedia secrecy.

This paper, a part of [6], proposes two space efficient secret sharing schemes for multimedia secrecy without recursion to ensure computational efficiency, where the first scheme is first shown in [6] and the second scheme can control the security of produced shares. Different from Shamir's and Parakh and Kak's secret sharing schemes, the proposed schemes ensure the integrity of the retrieved secret. The remainder of this paper is organized as follows. Section 2 reviews Parakh and Kak's secret sharing schemes and the corresponding examples that result in degree degeneration. In Section 3, the proposed space efficient

*Corresponding Author: Ya-Fen Chang; E-mail: cyf@nutc.edu.tw
DOI: 10.3966/160792642018111906025

secret sharing schemes without recursion are shown, and the corresponding examples are given. Security and efficiency analyses are made in Section 4. Finally, some conclusions are drawn in Section 5.

2 Review of Parakh and Kak's Secret Sharing Schemes and Counter Examples

Parakh and Kak proposed two secret sharing schemes for space efficiency. Parakh and Kak's original secret sharing scheme and mechanism to control the security of produced shares are briefly reviewed in Subsections 2.1 and 2.2, respectively. The corresponding examples that result in degree degeneration are reviewed in Subsection 2.3.

2.1 Parakh and Kak's Original Secret Sharing Scheme

Parakh and Kak's original secret sharing scheme is composed of two phases, dealing and reconstruction. The details are as follows.

2.1.1 Dealing Phase

In this phase, the secret holder/dealer generates n shares such that any k shares can reconstruct the secret S by the following steps.

Step 1: Divide the secret S into k-1 pieces, $s_1, s_2,\ldots, s_{k-1}$.

Step 2: Choose a large prime p, where $p > max(s_{max}, n)$ and $s_{max} = \max(s_i)$ for i=1, 2,…, k-1.

Step 3: Choose a number $a_1 \in Z_p$ randomly and uniformly.

Step 4: Create a polynomial $f_1(x) = a_1x + s_1 \bmod p$.

Step 5: Sample $f_{j-1}(x)$ at j points, $D_{j-1,1} = f_{j-1}(1)$, $D_{j-1,2} = f_{j-1}(2),\ldots, D_{j-1,j} = f_{j-1}(j)$, and generate a polynomial $f_j(x) = D_{j-1,j}x^j + D_{j-1,j-1}x^{j-1} +\ldots+ D_{j-1,1}x + s_j \bmod p$ for j = 2, 3,…, k-1.

Step 6: Sample $f_{k-1}(x)$ at n points, $D_1 = f_{k-1}(1), D_2 = f_{k-1}(2),\ldots, D_n = f_{k-1}(n)$.

Step 7: Distribute the share (i, D_i) to the shareholder U_i for i=1, 2,…, n.

2.1.2 Reconstruction Phase

When k shareholders want to reconstruct the secret S, the following steps are performed.

Step 1: Interpolate the collected k shares $(r', D_r') \in (i, D_i)$, where r=1, 2,…, k and i=1, 2,…, n, to reconstruct the (k-1)-th polynomial $f_{k-1}(x) = D_{k-2,k-1}x^{k-1} + D_{k-2,k-2}x^{k-2} +\ldots+ D_{k-2,1}x + s_{k-1} \bmod p$ and evaluate $s_{k-1} = f_{k-1}(0)$.

Step 2: Interpolate $(1, D_{j-1,1}), (2, D_{j-1,2}),\ldots, (j, D_{j-1,j})$ to extract all coefficients of $f_j(x)$, construct the (j-1)-th polynomial $f_{j-1}(x)$, and evaluate $s_{j-1} = f_{j-1}(0)$ for $j = k$-1, k-2,…, 2.

Step 3: Reconstruct the secret S with the k-1 pieces, $s_1, s_2,\ldots, s_{k-1}$.

2.2 Parakh and Kak's Mechanism to Control the Security of Produced Shares

Parakh and Kak's mechanism to control the security of produced shares is composed of two phases, dealing and reconstruction. The details are as follows.

2.2.1 Dealing Phase

In this phase, the secret holder/dealer generates n shares such that any k shares can reconstruct the secret S by the following steps, where the secret S is divided into m pieces instead of k-1 and m can be k-2, k-3, or 1 when the security of produced shares is further controlled.

Step 1: Divide the secret S into m pieces, $s_1, s_2,\ldots, s_m$.

Step 2: Choose a large prime p, where $p > max(s_{max}, n)$ and $s_{max} = \max(s_i)$ for i=1, 2,…, m.

Step 3: Choose k-m numbers $a_i \in Z_p$ randomly and uniformly for i=1, 2,…, k-m.

Step 4: Create a polynomial $f_1(x) = a_{k-m}x^{k-m} + a_{k-m-1}x^{k-m-1} +\ldots+a_1x + s_1 \bmod p$.

Step 5: Sample $f_{j-1}(x)$ at (k-m+j-1) points, $D_{j-1,1} = f_{j-1}(1)$, $D_{j-1,2} = f_{j-1}(2),\ldots, D_{j-1,k-m+j-1} = f_{j-1}(k\text{-}m\text{+}j\text{-}1)$, and generate a polynomial $f_j(x) = D_{j-1,k-m+j-1}x^{k-m+j-1} + D_{j-1,k-m+j-2}x^{k-m+j-2} +\ldots+ D_{j-1,1}x + s_j \bmod p$ for j = 2, 3,…, m.

Step 6: Sample $f_m(x) = D_{m-1,k-1}x^{k-1} + D_{m-1,k-2}x^{k-2} +\ldots+ D_{m-1,1}x + s_m \bmod p$ at n points, $D_1 = f_m(1), D_2 = f_m(2),\ldots, D_n = f_m(n)$.

Step 7: Distribute the share (i, D_i) to the shareholder U_i for i=1, 2,…, n.

2.2.2 Reconstruction Phase

When k shareholders want to reconstruct the secret S, the following steps are performed.

Step 1: Interpolate the collected k shares $(r', D_r') \in (i, D_i)$, where r=1, 2,…, k and i=1, 2,…, n, to reconstruct the m-th polynomial $f_m(x) = D_{m-1,k-1}x^{k-1} + D_{m-1,k-2}x^{k-2} +\ldots+ D_{m-1,1}x + s_m \bmod p$ and evaluate $s_m = f_m(0)$.

Step 2: Interpolate $(1, D_{j-1,1}), (2, D_{j-1,2}),\ldots, (k\text{-}m\text{+}j\text{-}1, D_{j-1,k-m+j-1})$ to extract all coefficients of $f_j(x)$, construct the (j-1)-th polynomial $f_{j-1}(x)$, and evaluate $s_{j-1} = f_{j-1}(0)$.

Step 3: Reconstruct the secret S with the m pieces, $s_1, s_2,\ldots, s_m$.

2.3 The Counter Examples

Tai et al. found that both of Parakh and Kak's secret sharing schemes suffer from degree degeneration and gave counter examples [5]. In the following, the counter examples given by Tai et al. are reviewed.

2.3.1 Review of the Counter Example of Parakh and Kak's Original Secret Sharing Scheme

In dealing phase, the secret dealer generates seven

shares by the following and any five shares can reconstruct the secret $S = 17280212$. First, the secret S is divided into four pieces s_1=17, s_2=28, s_3=02, and s_4=12. The secret dealer chooses a prime $p = 31$ such that $p > max(s_{max}, 7)$ and $s_{max} = \max(s_i)$ for i=1, 2,…, 4, and the secret dealer chooses a number a_1= 22 in Z_p. Polynomial $f_1(x) = a_1x + s_1 \bmod p = 22x + 17 \bmod 31$ is generated, and $f_1(x)$ is sampled at two points $D_{1,1} = f_1(1)$ =8 and $D_{1,2} = f_1(2) = 30$. Polynomial $f_2(x) = D_{1,2}x^2 + D_{1,1}x + s_2 \bmod p = 30x^2 + 8x + 28 \bmod 31$ is generated, and $f_2(x)$ is sampled at three points $D_{2,1} = f_2(1)$ =4, $D_{2,2} = f_2(2) = 9$, and $D_{2,3} = f_2(3)$ =12. Polynomial $f_3(x) = D_{2,3}x^3 + D_{2,2}x^2 + D_{2,1}x + s_3 \bmod p = 12x^3 + 9x^2 + 4x + 2 \bmod 31$ is generated, and $f_3(x)$ is sampled at four points $D_{3,1} = f_3(1)$ =27, $D_{3,2} = f_3(2) = 18$, $D_{3,3} = f_3(3)$ =16, and $D_{3,4} = f_3(4)$ =0. Polynomial $f_4(x) = D_{3,4}x^4 + D_{3,3}x^3 + D_{3,2}x^2 + D_{3,1}x + s_4 \bmod p = 0x^4 + 16x^3 + 18x^2 + 27x + 12 \bmod 31$ is generated, and $f_4(x)$ is sampled at seven points, $D_1 = f_4(1) = 11$, $D_2 = f_4(2) = 18$, $D_3 = f_4(3) = 5$, $D_4 = f_4(4) = 6$, $D_5 = f_4(5) = 24$, $D_6 = f_4(6) = 0$, and $D_7 = f_4(7) = 30$. These seven shares (1, 11), (2, 18), (3, 5), (4, 6), (5, 24), (6, 0), and (7, 30) are distributed to seven shareholders.

In reconstruction phase, only four shares (1, 11), (2, 18), (3, 5), and (4, 6) are obtained, but the secret still can be recovered because $D_{3,4} = f_3(4)$ =0. The details are as follows. Polynomial $f_4(x) = 0x^4 + 16x^3 + 18x^2 + 27x + 12 \bmod 31$ is constructed by interpolating (1, 11), (2, 18), (3, 5), and (4, 6), and $s_4 = f_4(0)$ =12 is evaluated. All coefficients of $f_4(x)$ are extracted to reconstruct $f_3(x) = 12x^3 + 9x^2 + 4x + 2 \bmod 31$ by interpolating (1, 27), (2, 18), (3, 16), and (4, 0), and $s_3 = f_3(0) = 2$ is evaluated. All coefficients of $f_3(x)$ are extracted to reconstruct $f_2(x) = 30x^2 + 8x + 28 \bmod 31$ by interpolating (1, 4), (2, 9), and (3, 12), and $s_2 = f_2(0) = 28$ is evaluated. All coefficients of $f_2(x)$ are extracted to reconstruct $f_1(x) = 22x + 17 \bmod 31$ by interpolating (1, 8) and (2, 30), and $s_1 = f_1(0) = 17$ is evaluated. At last, four pieces s_1=17, s_2=28, s_3=02, and s_4=12 are obtained with only four shares.

2.3.2 Review of the Counter Example of Parakh and Kak's Mechanism to Control the Security of Produced Shares

In dealing phase, the secret dealer generates seven shares by the following and any five shares can reconstruct the secret $S = 7901728$. First, the secret S is divided into two pieces s_1=790 and s_2=1728. The secret dealer chooses a prime $p = 1733$ such that $p > max(s_{max}, 7)$ and $s_{max} = \max(s_i)$ for i=1, 2, and the secret dealer chooses three numbers $a_1 = 105$, $a_2 = 61$, and a_3= 20. Polynomial $f_1(x) = a_3x^3 + a_2x^2 + a_1x + s_1 \bmod p = 20x^3 + 61x^2 + 105x + 790 \bmod 1733$ is generated, and $f_1(x)$ is sampled at four points $D_{1,1} = f_1(1)$ =976, $D_{1,2} = f_1(2)$ =1404, $D_{1,3} = f_1(3) = 461$, and $D_{1,4} = f_1(4) = 0$. Polynomial $f_2(x) = D_{1,4}x^4 + D_{1,3}x^3 + D_{1,2}x^2 + D_{1,1}x + s_2 \bmod p = 0x^4 + 461x^3 + 1404x^2 + 976x + 1728 \bmod 1733$ is generated, and $f_2(x)$ is sampled at seven points, $D_1 = f_2(1) = 1103$, $D_2 = f_2(2) = 853$, $D_3 = f_2(3) = 278$, $D_4 = f_2(4) = 411$, $D_5 = f_2(5) = 552$, $D_6 = f_2(6) = 1$, and $D_7 = f_2(7) = 1524$. At last, these seven shares (1, 1103), (2, 853), (3, 278), (4, 411), (5, 552), (6, 1), and (7, 1524) are distributed to seven shareholders.

In reconstruction phase, only four shares (1, 1103), (2, 853), (3, 278), and (7, 1524) are obtained, but the secret still can be recovered because $D_{1,4} = f_1(4)$ =0. The details are as follows. Polynomial $f_2(x) = 0x^4 + 461x^3 + 1404x^2 + 976x + 1728 \bmod 1733$ is constructed by interpolating (1, 1103), (2, 853), (3, 278), and (7, 1524), and $s_2 = f_2(0) = 1728$ is evaluated. All coefficients of $f_2(x)$ are extracted to reconstruct $f_1(x) = 20x^3 + 61x^2 + 105x + 790 \bmod 1733$ by interpolating (1, 976), (2, 1404), (3, 461), and (4, 0), and $s_1 = f_1(0) = 790$ is evaluated. At last, two pieces s_1=790 and s_2=1728 are obtained with only four shares.

3 The Proposed Space Efficient Secret Sharing Schemes Without Recursion

To ensure space efficiency and computational efficiency, we propose two secret sharing schemes without recursion, where the second one can control the security of produced shares. Different from Shamir's and Parakh and Kak's secret sharing schemes, the proposed schemes ensure the integrity of the retrieved secret by adopting a hash function. The proposed secret sharing scheme and mechanism to control the security of produced shares with examples are shown in Subsections 3.1 and 3.2, respectively.

3.1 Parakh and Kak's Original Secret Sharing Scheme

This scheme is composed of two phases: dealing phase and reconstruction phase. These two phases and the corresponding example are given as follows.

3.1.1 Dealing Phase

In this phase, the secret holder/dealer generates n shares such that any k shares can reconstruct the secret S by the following steps.

Step 1: Divide the secret S into k-1 pieces, s_1, s_2,…, s_{k-1}.

Step 2: Choose a large prime p, where $p > max(s_{max}, n)$ and $s_{max} = \max(s_i)$ for i=1, 2,…, k-1.

Step 3: Generate a collision-resistant one-way hash function $h(.)$, where $h(.)$'s computation result is in Z_p.

Step 4: Compute $a_{k-1} = h(I)$, where $I = S$. While a_{k-1} =0, $I = I$+1 and compute $a_{k-1} = h(I)$.

Step 5: Create a polynomial $f(x) = a_{k-1}x^{k-1} + a_{k-2}x^{k-2} +…+ a_1x + a_0 \bmod p$, where $a_i = s_{k-1-i}$ for i = 0, 1,…, k-2.

Step 6: Sample $f(x)$ at n points, $D_1 = f(ID_1)$, $D_2 = f(ID_2)$,…, $D_n = f(ID_n)$, where ID_i is the identifier of the

shareholder U_i for i = 1, 2,…, n and U_1, U_2,…, U_n are distinct shareholders.

Step 7: Distribute the share D_i to the shareholder U_i for i=1, 2,…, n.

3.1.2 Reconstruction Phase

When k shareholders want to reconstruct the secret S, the following steps are performed.

Step 1: Interpolate the collected k shares $D_r' \in D_i$, where r=1, 2,…, k and i=1, 2,…, n, to reconstruct the polynomial $f(x) = a_{k-1}x^{k-1} + a_{k-2}x^{k-2} + \ldots + a_1 x + a_0 \bmod p$.

Step 2: Reconstruct the secret S with the k-1 pieces, s_1, s_2,…, s_{k-1}, where $a_i = s_{k-1-i}$ for i = 0, 1,…, k-2.

Step 3: Check the integrity of S by checking whether $a_{k-1} = h(I)$, where $I = S$. If a_{k-1} =0, $I = I$+1 and compute $a_{k-1} = h(I)$.

3.1.3 The Example

Assume the secret S = 19213012. The secret holder/dealer will generate seven shares for users U_1, U_2,…, U_7, and the secret S can be reconstructed with any five shares, where U_i's identifier ID_i is i for i = 1, 2,…, n. The details are as follows.

In dealing phase, the secret holder/dealer performs the following.

Step 1: Divide the secret S into four pieces s_1= 19, s_2= 21, s_3= 30, and s_4= 12.

Step 2: Choose a large prime 31.

Step 3: Generate a collision-resistant one-way hash function h(.), where h(.)'s computation result is in Z_{31}.

Step 4: Compute $a_4 = h(I) = h$(19213012)=3.

Step 5: Create a polynomial $f(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1 x + a_0 \bmod p = 3x^4 + 19x^3 + 21x^2 + 30x + 12 \bmod 31$.

Step 6: Sample $f(x)$ at 7 points, $D_1 = f(ID_1) = f(1)$ =23, $D_2 = f(ID_2) = f(2)$ = 15, $D_3 = f(ID_3) = f(3)$ = 24, $D_4 = f(ID_4) = f(4)$ = 3, $D_5 = f(ID_5) = f(5)$ = 8, $D_6 = f(ID_6) = f(6)$ = 12, and $D_7 = f(ID_7) = f(7)$=29.

Step 7: Distribute the share D_i to the shareholder U_i for i=1, 2,…, 7.

In reconstruction phase, any five of (1, D_1), (2, D_2),…, (7, D_7) can reconstruct $f(x) = 3x^4 + 19x^3 + 21x^2 + 30x + 12 \bmod 31$ by interpolating them. Then the secret S is reconstructed such that S = 19213012. Then the integrity is checked by checking whether h(19213012)=3 holds or not.

3.2 The Proposed Secret Sharing Scheme with Controlling the Security of Produced Shares and the Corresponding Example

The second scheme is composed of two phases: dealing phase and reconstruction phase. These two phases and the corresponding example are given as follows.

3.2.1 Dealing Phase

In this phase, the secret holder/dealer generates n shares such that any k shares can reconstruct the secret S by the following steps, where the secret S is divided into m pieces instead of k-1 and m can be k-2, k-3, or 1 when the security of produced shares is further controlled.

Step 1: Divide the secret S into m pieces, s_1, s_2,…, s_m.

Step 2: Choose a large prime p, where $p > max(s_{max}, n)$ and $s_{max} = \max(s_i)$ for i=1, 2,…, m.

Step 3: Generate a collision-resistant one-way hash function h(.), where h(.)'s computation result is in Z_p.

Step 4: Choose k-m-1 numbers $a_i \in Z_p$ randomly and uniformly for i=k-2, k-3,…, m.

Step 5: Compute $a_{k-1} = h(I)$, where $I = S$. While a_{k-1} =0, $I = I$+1 and compute $a_{k-1} = h(I)$.

Step 6: Create a polynomial $f(x) = a_{k-1}x^{k-1} + a_{k-2}x^{k-2} + \ldots + a_1 x + a_0 \bmod p$, where $a_i = s_{m-i}$ for i = 0, 1,…, m-1.

Step 7: Sample $f(x)$ at n points, $D_1 = f(ID_1)$, $D_2 = f(ID_2)$,…, $D_n = f(ID_n)$, where ID_i is the identifier of the shareholder U_i for i = 1, 2,…, n and U_1, U_2,…, U_n are distinct shareholders.

Step 8: Distribute the share D_i to the shareholder U_i for i=1, 2,…, n.

3.2.2 Reconstruction Phase

When k shareholders want to reconstruct the secret S, the following steps are performed.

Step 1: Interpolate the collected k shares $D_r' \in D_i$, where r=1, 2,…, k and i=1, 2,…, n, to reconstruct the polynomial $f(x) = a_{k-1}x^{k-1} + a_{k-2}x^{k-2} + \ldots + a_1 x + a_0 \bmod p$.

Step 2: Reconstruct the secret S with the m pieces, s_1, s_2,…, s_m, where $a_i = s_{m-i}$ for i = 0, 1,…, m-1.

Step 3: Check the integrity of S by checking whether $a_{k-1} = h(I)$, where $I = S$. If a_{k-1} =0, $I = I$+1 and compute $a_{k-1} = h(I)$.

3.2.3 The Example

Assume the secret S = 100003 that is divided into two pieces. The secret holder/dealer will generate seven shares for users U_1, U_2,…, U_7, and the secret S can be reconstructed with any five shares, where U_i's identifier ID_i is i for i = 1, 2,…, n. The details are as follows.

In dealing phase, the secret holder/dealer performs the following.

Step 1: Divide the secret S into two pieces s_1= 100 and s_2= 003.

Step 2: Choose a large prime 113.

Step 3: Generate a collision-resistant one-way hash function h(.), where h(.)'s computation result is in Z_{113}.

Step 4: Choose two numbers a_3= 10 and a_2= 13.

Step 5: Compute $a_4 = h(I) = h$(010013100003)=3.

Step 6: Create a polynomial $f(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0 \bmod p = 3x^4 + 10x^3 + 13x^2 + 100x + 3 \bmod 113$.

Step 7: Sample $f(x)$ at 7 points, $D_1 = f(ID_1) = f(1) = 16$, $D_2 = f(ID_2) = f(2) = 44$, $D_3 = f(ID_3) = f(3) = 29$, $D_4 = f(ID_4) = f(4) = 98$, $D_5 = f(ID_5) = f(5) = 111$, $D_6 = f(ID_6) = f(6) = 0$, and $D_7 = f(ID_7) = f(7) = 108$.

Step 8: Distribute the share D_i to the shareholder U_i for i=1, 2,…, 7.

In reconstruction phase, any five of (1, D_1), (2, D_2),…, (7, D_7) can reconstruct $f(x) = 3x^4 + 10x^3 + 13x^2 + 100x + 3 \bmod 113$ by interpolating them. Then the secret S is reconstructed such that S = 100003. Then the integrity is checked by checking whether h(010013100003)=3 holds or not.

4 Security and Efficiency Analyses

In the proposed secret sharing schemes, one (k-1)-degree polynomial $f(x)$ is generated to produce n shares for n distinct shareholders. In the first scheme, the coefficients a_{k-2}, a_{k-3},…, a_1, a_0 are the pieces of the secret S while the coefficient $a_{k-1} = h(I) = h(S)$. In the second scheme, the coefficients a_{m-1},…, a_1, a_0 are the pieces of the secret S and a_{k-2}, a_{k-3},…, a_m are random numbers while the coefficient $a_{k-1} = h(I) = h(S)$, where h(.) is a one-way hash function with computation results in Z_p. In both schemes, if $a_{k-1} = 0$, $I = I+1$ and $a_{k-1} = h(I)$. The probability of $a_{k-1} = h(I) = h(S) = 0$ is $1/p$ because h(.) is a one-way hash function with computation results in Z_p. When $a_{k-1} = 0$, $I = I+1$ and the probability of $a_{k-1} = h(I) = 0$ is $1/p$. That is, the probability of $h(S) = h(S+1) = 0$ is $1/p^2$, and the probability of $h(S) = h(S+1) = h(S+2) = \ldots = h(S+t) = 0$ is $1/p^t$, where t denotes the times to consecutively obtain 0 with inputs S, $S+1$,…, $S+t$. However, p is a large prime such that the probability $1/p^t$ makes this case infeasible. Thus, via this approach, a_{k-1} will never be 0 such that the degree of $f(x)$ can be ensured to be (k-1).

In both schemes, the n shares are produced by sampling $f(x)$ at n points, where $D_1 = f(ID_1)$, $D_2 = f(ID_2)$,…, $D_n = f(ID_n)$. To reconstruct the polynomial $f(x)$ of degree (k-1), (k-1) shares are needed. That is, the proposed secret sharing schemes can resist degree degeneration, and the shared secret can be retrieved only when the number of involved shareholders is equal to or more than the legality threshold k. Moreover, to ensure the integrity of the recovered secret, the involved shareholders can check whether $a_{k-1} = h(I)$, where $I = S$. If $a_{k-1} = 0$, $I = I+1$ and compute $a_{k-1} = h(I)$.

As to computational efficiency, in both schemes, only one (k-1)-degree polynomial $f(x)$ is generated, and only $f(x)$ needs to be reconstructed when shareholders want to recover the secret S. The computational load in our schemes is much lighter than that of Parakh and Kak's because no recursive operation is needed.

5 Conclusions

This paper proposes two secret sharing schemes without recursion to ensure both space efficiency and computational efficiency, where the second scheme can control the security of produced shares. The proposed schemes also adopt a collision-resistant one-way hash function to check the recovered secret's integrity. Via the proposed secret sharing schemes, multimedia information can be shared among multiple users for security issues easily.

Acknowledgements

This work was supported in part by Ministry of Science and Technology under the Grants MOST Grants MOST 106-2221-E-034-006-, MOST 106-2410-H-025- 006-, MOST 106-2622-H-025-001-CC3, and MOST 107-2622-H-025-001-CC3.

References

[1] C. F. Hsu, Y. Liu, Q. Cheng, G. H. Cui, New Results on Multipartite Secret Sharing Matroids, *Journal of Internet Technology*, Vol. 11, No. 6, pp. 829-835, November, 2010.

[2] A. Shamir, How to Share a Secret, *Communications of the ACM*, Vol. 22, No. 11, pp. 612-613, November, 1979.

[3] A. Parakh, S. Kak, Space Efficient Secret Sharing for Implicit Data Security, *Information Sciences*, Vol. 181, No. 2, pp. 335-341, January, 2011.

[4] H. Krawczyk, Distributed Fingerprints and Secure Information Dispersal, *Proceedings of Twelfth Annual ACM Symposium on Principles of Distributed Computing*, New York, NY, 1993, pp. 207-218.

[5] W. L. Tai, Y. F. Chang, Y. F. Li, Comments on Space Efficient Secret Sharing Schemes for Implicit Data Security, *Proceedings of ICETA 2015-Fall*, Nagoya, Japan, 2015, pp. 79-85.

[6] W. L. Tai, Y. F. Chang, A Space Efficient Secret Sharing Scheme for Multimedia Secrecy Without Recursion, *Proceedings of CISC 2016*, Taichung, Taiwan, 2016, pp. 224-226.

Biographies

Wei-Liang Tai received the Ph.D. degree in computer science and information engineering from National Chung Cheng University, Taiwan, in 2008. He is currently Associate Professor, Department of Information Communications, Chinese Culture University. His main interests are in information security and forensics and multimedia signal processing. He is currently an Editor of KSII Transactions on Internet and Information Systems.

Ya-Fen Chang is a professor of Department of Computer Science and Information Engineering at National Taichung University of Science and Technology in Taiwan. She received her PhD degree in computer science and information engineering from National Chung Cheng University, Taiwan. Her current research interests include electronic commerce, information security, cryptography, mobile communications, image processing, and data hiding.