Computer Science and Information Engineering
National Chi Nan University

The Principle and Application of Secret Sharing

Dr. Justie Su-Tzu Juan

Lecture 3. Secret Sharing Scheme with Various Functions

§ 3.1 Verification and Detection

Slides for a Course Based on

1. P. Feldman, “*A practical scheme for non-interactive verifiable secret sharing*”, in Proceedings of 28th Foundations of Computer Science, pp.427-437, 1987.

2. *近代密碼學及其應用* by 賴溪松、韓亮、張真誠

§ 3.1 Verification and Detection

- **The drawbacks of Shamir's (*t*, *n*)-threshold scheme:**
  - Can't check validity of the shares from *D*
  - Can't verify validity of shares from other participants

- **Verification:** all participants can not check validity of the shares from the dealer. (**V**)
- **Detection:** participants can not verify validity of the shares from other participants. (**D**)

§ 3.1 Verification and Detection

- **Def**:

  1. p and q: large primes such that p divides $q - 1$.
  2. Z_p: a finite field with p elements.
  3. g: an element of order p of multiplicative group Z_q^*.
  4. D: dealer; $P = \{P_1, P_2, \ldots, P_n\}$; K: secret key.

§ 3.1 Verification and Detection

- **Feldman's scheme (1987):**
  - **Distribution:**

    Step 1. D publish n distinct nonzero elements $x_1, x_2, \ldots, x_n \in_r Z_p$.

    Step 2. D chooses $t-1$ elements $a_1, a_2, \ldots, a_{t-1} \in_r Z_p$ and construct

    $$f(x) = a_{t-1}x^{t-1} + \ldots + a_2x^2 + a_1x^1 + K \pmod{p}.$$

    Step 3. D distributes the share $S_i = f(x_i)$ to P_i.

    Step 4. D publishes $g, g^K \pmod{q}, g^{a_1} \pmod{q}, \ldots, g^{a_{t-1}} \pmod{q}$.

- **Note:** The number of publish data: $t + 3$. (p, q, g)

§ 3.1 Verification and Detection

- **Feldman's scheme (1987):**
  - **Checking (for Participant P_i):**

    1. Verify the authenticity of S_i:

    $$g^{S_i} =? \, (g^{a_{t-1}})^{x_i^{t-1}} (g^{a_{t-2}})^{x_i^{t-2}} \ldots (g^{a_1})^{x_i} (g^K) (\mathrm{mod}\, q).$$

    2. Detect the shares of S_j for any $1 \le j \le n$ by checking:

    $$g^{S_j} =? \, (g^{a_{t-1}})^{x_j^{t-1}} (g^{a_{t-2}})^{x_j^{t-2}} \ldots (g^{a_1})^{x_j} (g^K) (\mathrm{mod}\, q).$$

§ 3.1 Verification and Detection

- **Feldman's scheme (1987):**

Ex: K = 13, $(t, n) = (3, 5)$, $p = 17$, $f(x) = 13 + 10x + 2x^2$ (mod 17), and IDi = I

$q = 103$, $g = 8$ ($8^{17} = 1$ mod 103):

- **Distribution:**
  - $S_1 = f(1) = 8$; $S_2 = f(2) = 7$; $S_3 = f(3) = 10$; $S_4 = f(4) = 0$; $S_5 = f(5) = 11$.
- **Publish:**
  - $g = 8$, $g^K \pmod q = 30$, $g^{a_1} \pmod q = 93$, $g^{a_2} \pmod q = 64$.
- **Checking S_1 (for anyone):**
  - $g^{S_i} \pmod q = 8^8 \pmod{103} = 61$.
  - $(g^{a_2})^{x_1{}^2}(g^{a_1})^{x_1}(g^K) \pmod q = 64^1 \cdot 93^1 \cdot 30 \pmod{103} = 61$.

§ 3.1 Verification and Detection

- **Rabin's scheme (1994) and LHC (2002):**
  - **Distribution:**

    Step 1. D distributes the share S_i to P_i by any scheme.

    Step 2. For any $1 \le i \ne j \le n$, D select $X_{i,j}$ and $Y_{i,j} \in_r Z_p$ and calculate $Z_{i,j}$ such that $S_i = X_{i,j} + Y_{i,j} \cdot Z_{i,j} \pmod{p}$.

    Step 3. D distributes the checked keys $Z_{i,j}$, $(X_{j,i}, Y_{j,i})$ to P_i for $1 \le j \le n$ and $j \ne i$.

- **Note:** The number of extra data for each participant: $3(n-1)$.

§ 3.1 Verification and Detection

- **Rabin's scheme (1994) and LHC (2002):**
  - **Checking (for Participant P_i):**

    1. Detect the shares of S_j for any $1 \leq j \leq n$ by checking:

$$S_j =?\ X_{j,\,i} + Y_{j,\,i} \cdot Z_{j,\,i} \pmod{p},$$

where $X_{j,\,i}$, $Y_{j,\,i}$ from P_i; $Z_{j,\,i}$ from P_j.

- **Note:** For P_j, $(S_j, Z_{j,\,i})$ can not help to obtain another point $(S_j^*, Z_{j,\,i}^*)$ such that $S_j^* = X_{j,\,i} + Y_{j,\,i} \cdot Z_{j,\,i}^* \pmod{p}$. It is equivalent to knowing only one point in a linear equation but not being able to find another point on the same line.

§ 3.1 Verification and Detection

- **Rabin's scheme (1994) and LHC (2002):**

Ex: K = 13, $(t, n) = (3, 5)$, $p = 17$, $f(x) = 13 + 10x + 2x^2$ (mod 17), and IDi = I

– **Distribution:**

- $S_1 = f(1) = 8$; $S_2 = f(2) = 7$; $S_3 = f(3) = 10$; $S_4 = f(4) = 0$; $S_5 = f(5) = 11$.
- For P_1, D select $X_{2,1} = 4$, $X_{3,1}$, $X_{4,1}$, $X_{5,1}$ and $Y_{2,1} = 2$, $Y_{3,1}$, $Y_{4,1}$, $Y_{5,1}$ and calculate $Z_{1,2}$, $Z_{1,3}$, $Z_{1,4}$, $Z_{1,5}$, such that $S_1 = X_{1,j} + Y_{1,j} \cdot Z_{1,j}$ (mod 13).
- For P_2, D select $X_{1,2}$, $X_{3,2}$, $X_{4,2}$, $X_{5,2}$ and $Y_{1,2}$, $Y_{3,2}$, $Y_{4,2}$, $Y_{5,2}$ and calculate $Z_{2,1}$, $Z_{2,3}$, $Z_{2,4}$, $Z_{2,5}$, such that $S_2 = X_{2,j} + Y_{2,j} \cdot Z_{2,j}$ (mod 13).
- ...
- Where D distributes the checked keys $(X_{2,1}, Y_{2,1}) = (4, 2)$ to P_1; and distributes $Z_{2,1} = 10$ to P_2;

§ 3.1 Verification and Detection

- **Rabin's scheme (1994) and LHC (2002):**

  Ex: K = 13, $(t, n) = (3, 5)$, $p = 17$, $f(x) = 13 + 10x + 2x^2 \pmod{17}$, and ID$i = I$

  – **Distribution:**

  - Where D distributes the checked keys $(X_{2,1}, Y_{2,1}) = (4, 2)$ to P_1;

    and distributes $Z_{2,1} = 10$ to P_2;

  – **Checking S_2 for P_1:**

  - P_1 detect the shares of S_2 by checking:

    $$S_2 = 7 \text{ (from } P_2)$$

    $$X_{2,1} + Y_{2,1} \cdot Z_{2,1} = 4 + 2 \cdot 10 = 7 \pmod{17},$$

    where $X_{2,1}$, $Y_{2,1}$ from P_1; $Z_{2,1}$ from P_2.

Computer Science and Information Engineering
National Chi Nan University

The Principle and Application of Secret Sharing

Dr. Justie Su-Tzu Juan

Lecture 3. Secret Sharing Scheme with Various Functions

§ 3.2 Multi-Secret Sharing Scheme

Slides for a Course Based on

1. C.-C. Yang, T.-Y. Chang and M.-S. Hwang, "A (t, n) multi-secret sharing scheme", Applied Mathematics and Computation, pp.483-490, 2004.

2. *近代密碼學及其應用* by 賴溪松、韓亮、張真誠

§ 3.2 Multi-Secret Sharing Scheme

- **The drawbacks of Shamir's (*t*, *n*)-threshold scheme:**
  - Shares held by the participants are used only once.

- **Multi-use:** If we want to share a new secret, the dealer does not need to redistribute new shares to each participant. (M)
- Also called **Multi-Secret Sharing Scheme** (**MSSS**), **Online SSS.**

§ 3.2 Multi-Secret Sharing Scheme

- **Def**: **Two-variable one-way hash function ($f_{hash}(r, s)$)**: In which s and r are two numbers, and $f_{hash}(r, s)$ will be a bit string with a fixed length. It has the following properties :
  - (1) Given r and s, it is easy to compute $f_{hash}(r, s)$.
  - (2) Given s and $f_{hash}(r, s)$, it is hard to compute r.
  - (3) Having no information of s, it is hard to compute $f_{hash}(r, s)$ for any r.
  - (4) Given s, it is hard to find two different values r_1 and r_2 such that $f_{hash}(r_1, s) = f_{hash}(r_2, s)$.
  - (5) Given r and $f_{hash}(r, s)$, it is hard to compute s.
  - (6) Given pairs of $(r_i, f_{hash}(r_i, s))$, it is hard to compute $f_{hash}(r', s)$ for $r' \neq r_i$.

§ 3.2 Multi-Secret Sharing Scheme

- **Yang et al.'s (t, n) multi-secret sharing scheme (2004):**
  - **Step 0.** D randomly chooses n shares $s_1, s_2, \ldots, s_n$ and sends s_i to P_i.
  - **Distribution (k secret ($K_1, K_2, \ldots, K_k$)):**
    - **If $k \le t$**

      Step 1. Choose a prime p and construct $(t-1)$th degree polynomial $g(x)$ mod p,

      where $0 < K_1, K_2, \ldots, K_k, a_1, a_2, \ldots, a_{t-k} < p$ as follows:

      $g(x) = K_1 + K_2x^1 + \ldots + K_kx^{k-1} + a_1x^k + a_2x^{k+1} + \ldots + a_{t-k}x^{t-1} \pmod{p}$.

      Step 2. Compute $y_i = g(f_{\text{hash}}(r, s_i))$ mod p for $i = 1, 2, \ldots, n$.

      Step 3. Publish $(r, y_1, y_2, \ldots, y_n)$.

- **Note:** The number of publish data: $n + 1$.

§ 3.2 Multi-Secret Sharing Scheme

- **Yang et al.'s (t, n) multi-secret sharing scheme (2004):**
  - **Step 0.** D randomly chooses n shares $s_1, s_2, \ldots, s_n$ and sends s_i to P_i.
  - **Distribution (k secret ($K_1, K_2, \ldots, K_k$)):**
    - **If $k > t$**

      Step 1. Choose a prime p and construct $(k - 1)$th degree polynomial $g(x)$ mod p, where $0 < K_1, K_2, \ldots, K_k < p$ as follows:

      $$g(x) = K_1 + K_2x^1 + \ldots + K_kx^{k-1} \pmod{p}.$$

      Step 2. Compute $y_i = g(f_{\text{hash}}(r, s_i))$ mod p for $i = 1, 2, \ldots, n$.

      Step 3. Compute $g(i)$ mod p for $i = 1, 2, \ldots, k - t$.

      Step 4. Publish $(r, g(1), g(2), \ldots, g(k - t), y_1, y_2, \ldots, y_n)$.

- **Note:** The number of publish data: $n + k - t + 1$.

§ 3.2 Multi-Secret Sharing Scheme

- **Yang et al.'s (t, n) multi-secret sharing scheme (2004):**
  - **Reconstruction** (Collect t pairs of $(f_{\text{hash}}(r, s_i), y_i)$, say $1 \le i \le t$ W.L.O.G.)**:**
    - **If $k \le t$**

Step 1. Using Lagrange Interpolation Formula:

$$g(x) = \sum_{i=1}^{t}\left(y_i \prod_{j=1, j\neq i}^{t} \frac{x - f_{\text{hash}}(r, s_j)}{f_{\text{hash}}(r, s_i) - f_{\text{hash}}(r, s_j)}\right) \bmod q$$

$$= K_1 + K_2 x^1 + \ldots + K_k x^{k-1} + a_1 x^k + a_2 x^{k+1} + \ldots + a_{t-k} x^{t-1} \pmod p.$$

Step 2. Get k secrets $(K_1, K_2, \ldots, K_k)$.

§ 3.2 Multi-Secret Sharing Scheme

- **Yang et al.'s (t, n) multi-secret sharing scheme (2004):**
  - **Reconstruction** (Collect t pairs of $(f_{\text{hash}}(r, s_i), y_i)$, say $1 \le i \le t$ W.L.O.G.)**:**
    - **If $k > t$**

Step 1. Get $(i, g(i))$ for $1 \le i \le k - t$.

Step 2. Using Lagrange Interpolation Formula:

$$g(x) = \sum_{i=1}^{t}\left(y_i \prod_{l=1}^{k-t} \frac{x-l}{f_{\text{hash}}(r,s_i)-l} \prod_{j=1, j\neq i}^{t} \frac{x-f_{\text{hash}}(r,s_j)}{f_{\text{hash}}(r,s_i)-f_{\text{hash}}(r,s_j)}\right)$$

$$+ \sum_{i=1}^{k-t}\left(g(i) \prod_{j=1, j\neq i}^{k-t} \frac{x-j}{i-j} \prod_{l=1}^{t} \frac{x-f_{\text{hash}}(r,s_l)}{i-f_{\text{hash}}(r,s_l)}\right) \pmod{q}$$

$$= K_1 + K_2 x^1 + \ldots + K_k x^{k-1} \pmod{p}.$$

Step3. Get k secrets $(K_1, K_2, \ldots, K_k)$.

§ 3.2 Multi-Secret Sharing Scheme

- **Yang et al.'s (t, n) multi-secret sharing scheme (2004):**
  - When the dealer shares a new secret, the dealer only need to publishes new (r, y_1, y_2, ..., y_n) when $k \le t$, or (r, $g(1)$, $g(2)$, ..., $g(k - t)$, y_1, y_2, ..., y_n) when $k > t$. The dealer need not redistribute shares to each participant.

- **Observation**: When $k > 1$, it is not perfect!

§ 3.2 Multi-Secret Sharing Scheme

- **Harn's (t, n) multi-secret sharing scheme (1995): Using DSA**
- **Def: by dealer D**

1. q: large prime > 800 bits.
2. p: large prime > 160 bits such that p divides $q - 1$.
3. $a_i \in Z_p^*$ for $0 \le i \le t - 1$, and

$$f(x) = a_{t-1}x^{t-1} + \ldots + a_2x^2 + a_1x^1 + a_0 \pmod{p}.$$

4. $g_i \in Z_p^*$ for $0 \le i \le k$ and $g_i = h_i^{(q-1)/p} \bmod q$, for any integer h_i.

   Note, g_i will be an generator with order p of multiplicative group Z_q^*.
5. Secret $K_i = g_i^{a_0} \bmod q$ $(= g_i^{f(0)} \bmod q)$.

§ 3.2 Multi-Secret Sharing Scheme

- **Harn's (t, n) multi-secret sharing scheme (1995): Using DSA**
  - **Distribution (secret K_i):**

    Step 1. Send $S_i = f(x_i)$ to P_i; x_i is the ID of P_i, where

    $$f(x) = a_{t-1}x^{t-1} + \ldots + a_2x^2 + a_1x^1 + a_0 \pmod{p}.$$

    Step 2. D publish (t, k, p, q, g_i).

- **Note:** The number of publish data: $4 + k$.

§ 3.2 Multi-Secret Sharing Scheme

- **Harn’s (*t*, *n*) multi-secret sharing scheme (1995): Using DSA**
  - **Reconstruction** (Collect t pairs of (x_j, $S_{i,j} = g_i^{S_j}$ mod q), say $1 \le j \le t$ W.L.O.G.**):**

    Step 1. Using Lagrange Interpolation Formula:

$$\boldsymbol{K_i} = \prod_{j=1}^{t} S_{i,j}^{\prod_{l=1,l\neq j}^{t}\frac{-x_j}{x_j-x_l}\bmod p} \bmod q = g_i^{\sum_{j=1}^{t} S_j \prod_{l=1,l\neq j}^{t}\frac{-x_j}{x_j-x_l}\bmod p} \bmod q$$

$$= g_i^{f(0)} \bmod q.$$

§ 3.2 Multi-Secret Sharing Scheme

- **Harn's (t, n) multi-secret sharing scheme (1995): Using DSA**
- **Analysis**: Due to the security of DLP, handing over $S_{i,j} = g_i^{S_j} \bmod q$ will not reveal any information about S_j; at the same time, getting $K_i = g_i^{a_0} \bmod q$ will not know any information about a_0.

Computer Science and Information Engineering
National Chi Nan University

The Principle and Application of Secret Sharing

Dr. Justie Su-Tzu Juan

Lecture 3. Secret Sharing Scheme with Various Functions

§ 3.3 No Dealer

Slides for a Course Based on
近代密碼學及其應用 **by 賴溪松、韓亮、張真誠**

§ 3.3 No Dealer

- **The drawbacks of Shamir's (*t*, *n*)-threshold scheme:**
  - In general business and large-scale secret sharing, it is very difficult to find a trustworthy person to be the dealer.

- **No dealer:** Secret sharing scheme without dealer assistance.

§ 3.3 No Dealer

- **Ingemarsson and Simmons's scheme: A (t, n) secret sharing scheme without the assistance of a trusted party (1991):**
  - **Distribution (secret K):**

    Step 1. Every participant P_i select a key K_i for himself.

    Step 2. Set secret key $K = K_1 + K_2 + \ldots + K_n$

    Step 3. Every participant P_i:

    3.1. Use (t, n – 1)-TS to construct

    $K_{i,1}, K_{i,2}, \ldots, K_{i,i-1}, K_{i,i+1}, K_{i,n}$, for his key K_i.

    3.2. Send $K_{i,j}$ to P_j.

§ 3.3 No Dealer

- **Ingemarsson and Simmons's scheme: A (t, n) secret sharing scheme without the assistance of a trusted party (1991):**
  - **Reconstruction (**Collect t pairs of (K_i, $K_{j,\,i}$), say $1 \le i \le t$, $t + 1 \le j \le n$, W.L.O.G.**):**

    Step 1. For $t + 1 \le j \le n$, reconstruct K_j by $K_{j,\,1}$, $K_{j,\,2}$, …, $K_{j,\,t}$.

    Step 2. Get key $K = K_1 + K_2 + \ldots + K_n$

§ 3.3 No Dealer

- **Ingemarsson and Simmons's scheme: A (t, n) secret sharing scheme without the assistance of a trusted party (1991):**

  Ex: (t, n) = (2, 3)

  – **Distribution:**
    - P_1 get K_1, $K_{2,1}$, $K_{3,1}$
      P_2 get K_2, $K_{1,2}$, $K_{3,2}$
      P_3 get K_3, $K_{1,3}$, $K_{2,3}$

  – **Reconstruction by P_1 and P_2:**
    - P_1 and P_2 calculate K_3 by $K_{3,1}$ (from P_1) and $K_{3,2}$ (from P_2)
    - Get key $K = K_1 + K_2 + K_3$

Computer Science and Information Engineering
National Chi Nan University

The Principle and Application of Secret Sharing

Dr. Justie Su-Tzu Juan

Lecture 4. The Geometric Approach for Sharing Secrets

§ 4.1 A (k, n)-Threshold Scheme Based on a Hyperspherical Function (HS-TS)

Slides for a Course Based on

T.-C. Wu and W.-H. He, "A geometric approach for sharing secrets", Computer & Security, pp.135-145, 1995

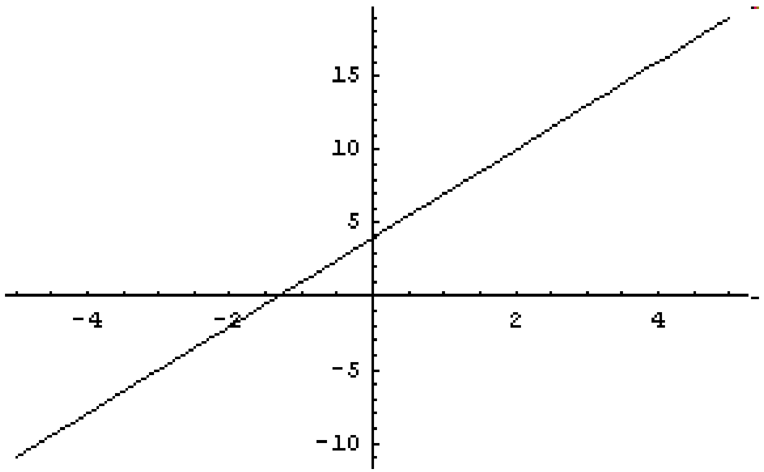
§ 4.1 A (*k*, *n*)-Threshold Scheme Based on a Hyperspherical Function (HS-TS)

- **Def:**
  - Let $\mathcal{K}$ be the master key space and $\mathcal{S}$ be the share space. The *information rate* of the secret sharing scheme is defined as $\log_2|\mathcal{K}| / \log_2|\mathcal{S}|$.
  - A secret sharing scheme is *perfect* if any set of participants in the prohibited structure obtains no information regarding the secret.
  - Secret sharing schemes are classified into the following types:
    - Type I: A secret sharing scheme for the *access structure* Γ: $\Delta = 2^P - \Gamma$.
    - Type II: A secret sharing scheme for the *prohibited structure* Δ: $\Gamma = 2^P - \Delta$.
    - Type III: A secret sharing scheme for the *mixed structure* (Γ, Δ): $(\Gamma \cup \Delta) \subseteq 2^P$

§ 4.1 A (k, n)-Threshold Scheme Based on a Hyperspherical Function (HS-TS)

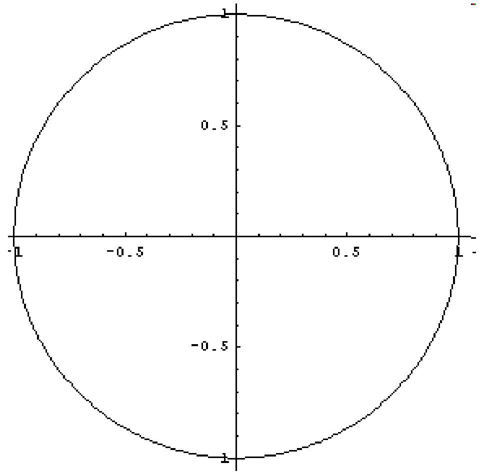
- **Simple geometric properties:**
  - 1. $(x_1, y_1), (x_2, y_2)$ -------------- $y = ax + b$

    a (2, n)-threshold scheme.



  - 2. $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ ----- $(x_1 - a_1)^2 + (x_2 - a_2)^2 = s$

    a (3, n) threshold scheme.



§ 4.1 A (k, n)-Threshold Scheme Based on a Hyperspherical Function (HS-TS)

- **Simple geometric properties:**
  - 3. $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)$ ----- $(x_1 - a_1)^2 + (x_2 - a_2)^2 + (x_3 - a_3)^2 = s$

    a (4, n) threshold scheme.

  - 4. Extend 2 and 3 to k items:
    - Given any k points, which don't lie on $(k - 2)$-dimensional space, can uniquely determine $(a_1, a_2, \ldots, a_{k-1})$ and s, such that:

$$\sum_{i=1}^{k-1}(x_i - a_i)^2 = s.$$

Device a (k, n) threshold scheme.