Chapter 6

Scheme for $(n-1, n)$ threshold

6.1 Introduction

In this section, we present our method to construct an $(n-1, n)$ secret sharing scheme based on the modified visual cryptography. In this scheme, every bit is expanded to $\lceil \frac{n}{2} \rceil$ many bits.

6.2 A new scheme

Let the participants be $\{P_1, P_2, P_3, \ldots, P_n\}$. In this case, the access structure consists of all the $n-1$ participants, namely:

$$\Gamma = \bigcup_{i=1}^{n} P_1 P_2 \ldots P_{i-1} \widehat{P}_i P_{i+1} \ldots P_{n-1} P_n$$

Here the $\widehat{P_i}$ indicate the absence of the participants P_i in the set. The complete elements can be listed as follows:

$$\begin{array}{lccccccccc}
1. & \widehat{P_1} & P_2 & P_3 & P_4 & \dots & P_{n-2} & P_{n-1} & P_n \\
2. & P_1 & \widehat{P_2} & P_3 & P_4 & \dots & P_{n-2} & P_{n-1} & P_n \\
3. & P_1 & P_2 & \widehat{P_3} & P_4 & \dots & P_{n-2} & P_{n-1} & P_n \\
4. & P_1 & P_2 & P_3 & \widehat{P_4} & \dots & P_{n-2} & P_{n-1} & P_n \\
\vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\
n. & P_1 & P_2 & P_3 & P_4 & \dots & P_{n-2} & P_{n-1} & \widehat{P_n}
\end{array}$$

We can see that the first two sets differ in P_1 and P_2; the next two sets differ in P_3 and P_4; and so on. If we combine these sets pairwise, if n is even, there are exactly $\frac{n}{2}$ pairs of sets and if n is odd, there are $\lfloor \frac{n}{2} \rfloor$ many pairs and one set left out. Let the secret be $B = B_1B_2B_3 \dots B_t$. Our scheme will generate n shares for each bit B_i of the secret.

6.3 Algorithm for sharing one bit among n shares

The following Algorithm describes how to share a single bit b among n shares.

Algorithm 6.1 (Sharing one bit among n shares)
Input: A binary bit $b \in \{0, 1\}$
Output: The n *shares* $S_1, S_2, \dots, S_n$*, where,*
each S_i *is of length* $\lceil \frac{n}{2} \rceil$ *bits.*

Step 1. *Let* $S_{i,j}$ *denote the* j^{th} *bit of* S_i

For $j = 1$ *to* $\lfloor \frac{n}{2} \rfloor$ *do*

$x = b$

For $i = 1$ *to* n *do*

if $(i \neq 2j-1$ *AND* $i \neq 2j)$ {

Generate a random number $r \in \{0, 1\}$

$S_{i,j} = r$

$x = x \oplus r$

}

$S_{2j-1,j} = S_{2j,j} = x$

Step 2. *If* (n *is odd*) *then* { \\ *Here* $j = \lceil \frac{n}{2} \rceil$

$x = b$

For $i = 1$ *to* $n - 2$ *do*

Generate a random number $r \in \{0, 1\}$

$S_{i,j} = r$

$x = x \oplus r$

$S_{n-1,j} = x$

} \\ *Note that in this case,* $S_{n,j}$ *is unknown*

Step 3. *The shares are* $S_1, S_2, \ldots, S_n$

Algorithm 6.2 (Recover the shared secret bit b)

Input: $n - 1$ *shares* $S_1 S_2 \ldots S_{j-1} S_{j+1} \ldots S_n$, 2

each of length $\lceil \frac{n}{2} \rceil$ *bits*

Observe that S_j *is the missing share.* 4

Output: The shared secret bit b

Step 1. *Let* $c = \lceil \frac{j}{2} \rceil$ *and* $x = 0$

For $k = 1$ *to* n *do*

if $(k \neq j)$ $x = x \oplus S_{k,c}$

$b = x$

Step 2. *The shared secret bit is recovered as* b

Lemma 6.1

The above scheme is a $(n-1, n)$ *threshold secret sharing scheme, in which the size of a share is* $\lceil \frac{n}{2} \rceil$ *bits.*

Proof: It is easy to observe the following from Algorithm 6.1.

1. For each $j \in \{1, \ldots, \lfloor \frac{n}{2} \rfloor\}$, the Step 1. of the algorithm generates $n-2$ random bits and assigns one each to $S_{i,j}$ for $i \in \{1, \ldots, n\} \setminus \{2j-1, 2j\}$.

2. The final value of x computed in the inner for loop is $x = b \oplus S_{1,j} \oplus \ldots \oplus S_{2j-2,j} \oplus S_{2j+1,j} \oplus \ldots \oplus S_{n,j}$

3. This value of x is assigned to $S_{2j-1,j}$ and $S_{2j,j}$.
   So, $S_{1,j} \oplus \ldots \oplus S_{2j-1,j} \oplus S_{2j+1,j} \oplus \ldots \oplus S_{n,j} = b$
   and $S_{1,j} \oplus \ldots \oplus S_{2j-2,j} \oplus S_{2j,j} \oplus \ldots \oplus S_{n,j} = b$

4. If n is odd, Step 2 of the algorithm generates $n-2$ random bits and assigns one each to $S_{i,j}$ for $i \in \{1, \ldots, n-2\}$.
   The final value of x computed in the for loop is $x = b \oplus S_{1,j} \oplus \ldots \oplus S_{n-2,j}$

5. This value of x is assigned to $S_{n-1,j}$.
   So, $S_{1,j} \oplus \ldots \oplus S_{n-1,j} = b$

Algorithm 6.3 (Sharing a secret among n shares)

Input: A binary string $B = B_1B_2 \dots B_t$ *of length* t

Output : The n *shares* $S_1, S_2, \dots, S_n$*, where,*

each S_i *is of length* $\lceil \frac{n}{2} \rceil$ *times* t*.*

Step 1. *For* $i = 1$ *to* n *do*

Initialize S_i *to NULL*

Step 2. *For* $i = 1$ *to* t *do*

Compute the n *shares corresponding to* B_i *using Algorithm 6.1 and append to the corresponding* S_j*, for* $j = \{1, \dots, n\}$*.*

Algorithm 6.4 (Recover the shared secret)

Input: $n-1$ *shares* $S_1S_2 \dots S_{j-1}S_{j+1} \dots S_n$*,*

each of length t *times* $\lceil \frac{n}{2} \rceil$

Observe that S_j *is the missing share.*

Output: The shared secret $B = B_1B_2 \dots B_t$

Step 1. *Let* $S_j^{(1)}, S_j^{(2)}, \dots S_j^{(t)}$ *be the consecutive bits of length* $\lceil \frac{n}{2} \rceil$ *in* S_j*, for* $j \in \{1, \dots, n\}$

For $i = 1$ *to* t *do*

Recover the secret bit B_i *by using Algorithm 6.2 with input* $S_j^{(i)}$*, for* $j \in \{1, \dots, n\}$

Step 2. *The shared secret is* $B = B_1B_2 \dots B_t$

Example 6.1

Let a (4, 5) threshold secret sharing scheme be constructed for the secret $B =$ *10111 10111 10111 (which corresponds to "www").*

Here n = 5, so each bit will be expanded to 3 bits. The random bits generated by the Algorithm 6.3, and assigned at various places in the shares are as follows: (the $*$ indicates NULL bit and - indicates an unknown bit)

Table 6.1: Random bits assigned in the shares by Algorithm 6.1.

S_1	$*10*01*10*00*10*10*01*10*10*11*10*01*01*10*00$
S_2	$*10*00*10*11*01*11*10*10*00*01*01*10*10*01*11$
S_3	$1*10*10*00*01*10*10*11*01*10*11*01*10*10*01*1$
S_4	0**1**0**0**1**0**1**0**1**0**1**0**1**0**1**
S_5	01–01–10–01–01–10–01–11–11–01–00–11–00–00–10–

The bit values at the NULL positions are evaluated and the final shares are as seen in Table 6.2.

Table 6.2: Final Shares computed by Algorithm 6.1.

S_1	010101010100110010101110010111110001001110000
S_2	010100010111101011110110000101101010010101011
S_3	101011010010111011001100111011100101001000101
S_4	000110011010111011100001110010100000101000101
S_5	01–01–10–01–01–10–01–11–11–01–00–11–00–00–10–

Suppose we want to reconstruct the secret from 1^{st}, 3^{rd}, 4^{th} and 5^{th} shares. If we compute $S_1 \oplus S_3 \oplus S_4 \oplus S_5$, we get, result as 10-01-11-11-10-11-01-10-10-10-11-01-10-11-100. Here 2^{nd} share is missing. So every first bit in the block of 3 bits are selected

as : 10111 10111 10111

Suppose we want to reconstruct the secret from 1^{st}, 2^{nd}, 3^{rd}, and 4^{th}. If we compute $S_1 \oplus S_2 \oplus S_3 \oplus S_4$, we get, result as

$$101100001011011001110101011011011110111011011$$

Here 5^{th} share is missing. So every third bit in the block of 3 bits are selected as : 10111 10111 10111

6.4 Concluding remarks

We have now presented an $(n-1, n)$-threshold secret sharing scheme, in which the size of a share is $\lceil \frac{n}{2} \rceil$ times the size of the secret.