A Software Implementation of Shamir' s Secret Sharing Scheme

NG Cheuk Hin, YUEN Ho Cheung and YUNG Cho Fu Aaron

Advised by Prof. Cunsheng DING

Introduction

Objectives

We aim to design a webpage that allows people to try secret sharing scheme. The scheme that we are implementing is the Shamir' s Secret Sharing Scheme. It mainly divides into Secret Splitting (Encryption) and Secret Recovering (Decryption).

Shamir's Secret Sharing Scheme

- Introduced by Adi Shamir.
- Divides a secret into n parts, giving each participant its own unique part, where only some, any k out of n participants, are necessary for recovering the secret.
- Secret: An element $S \in Z_p$, where p is prime number. This value may represent ASCII value in order to give us words and symbols after certain conversion.
- Participants: P1, P2, ... , Pn .
- Computing and distributing the shares: A dealer first chooses n distinct nonzero elements of Z_p, denoted x_i, $1 \le i \le n < p$. The dealer then gives xi to participants Pi. The values x_i are public. Then the dealer chooses (independently at random) k - 1 elements $a_1, a_2, \dots, a_{k-1} \in Z_p$. For each $1 \le i \le n$, the dealer computes $y_i = f(x_i)$, where

$$f(x) = S + \sum_{j=1}^{k-1} a_j x^j \bmod p$$

The dealer then gives y_i, which is $f(x_i)$ to participants Pi.

- Reconstruction: Suppose that the function $y = g(x)$ is known at the k points $(x_1,y_1), (x_2,y_2), \dots, (x_k,y_k)$, where $x_1 < x_2 \dots < x_k$,

$$\begin{aligned} S + a_1 x_1 + a_2 x_1^2 + \dots + a_{k-1} x_1^{k-1} &= y_1 \\ S + a_1 x_2 + a_2 x_2^2 + \dots + a_{k-1} x_2^{k-1} &= y_2 \\ \dots \\ S + a_1 x_k + a_2 x_k^2 + \dots + a_{k-1} x_k^{k-1} &= y_k \end{aligned}$$

Consider the equations above, we can treat them as creating a polynomial $f_{k-1}(x_i) = y_i$ using the idea of Lagrange Polynomial, which is one of the methods to recover a polynomial by given sets of x_i and y_i

$$f(x) = \sum_{i=1}^{k} y_i \prod_{1 \le i \le k,\ i \ne j} \frac{x - x_j}{x_i - x_j} \bmod p$$

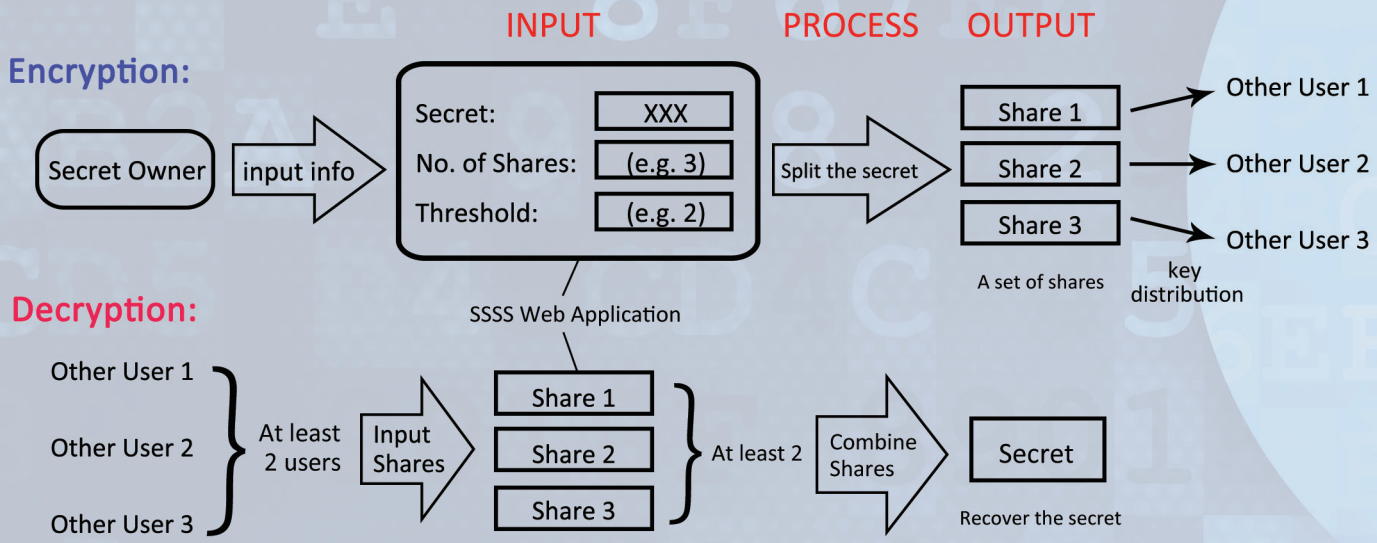
And thus, the secret can be computed by S = f(0), since only the constant term S is left.

$$S = f(0) = \left(\sum_{i=1}^{k} y_i \prod_{1 \le i \le k,\ i \ne j} \left(\frac{x_j}{x_j - x_i}\right) \bmod p\right) \bmod p$$

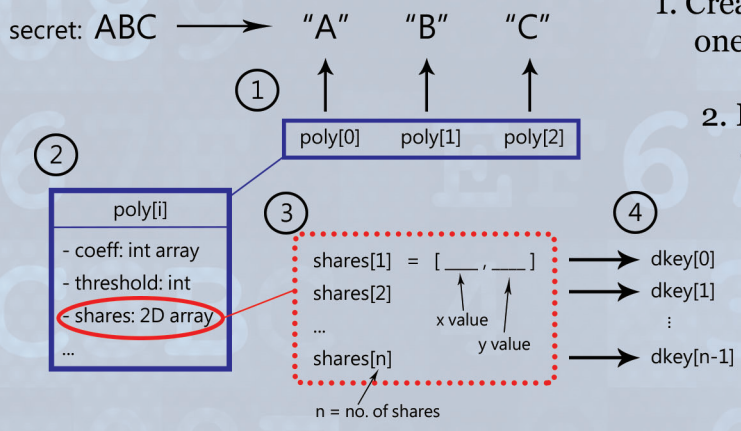
We can use geometry to explain. In Cartesian coordinate system, we can define a straight line by two points, define a parabola by three points, and define a cubic curve by four points and so on. Generally speaking, it takes k points to define a polynomial of degree k-1.
Shamir' s secret sharing uses the same principle. It generates k keys (parts) since k points to recover the polynomial of degree k-1 where the secret is inside the polynomial.

Methodology

System Architecture



Secret Splitting (Encryption)



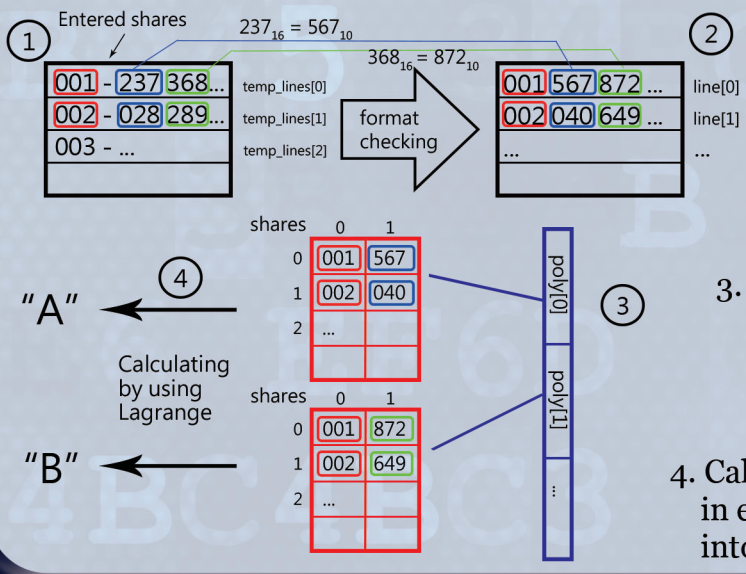
1. Create an array for representing polynomial and one polynomial represents one character

2. Each polynomial contains various properties e.g. coefficients, threshold, shares, etc.

3. A 2D array "shares" stores the x inputs and y outputs, where y = f(x) mod p.

4. In distribution of shares, each key contains the x input and all the corresponding y outputs.

Secret Recoveering (Decryption)



1. Separates the keys and stores them in an array, followed by format checking.

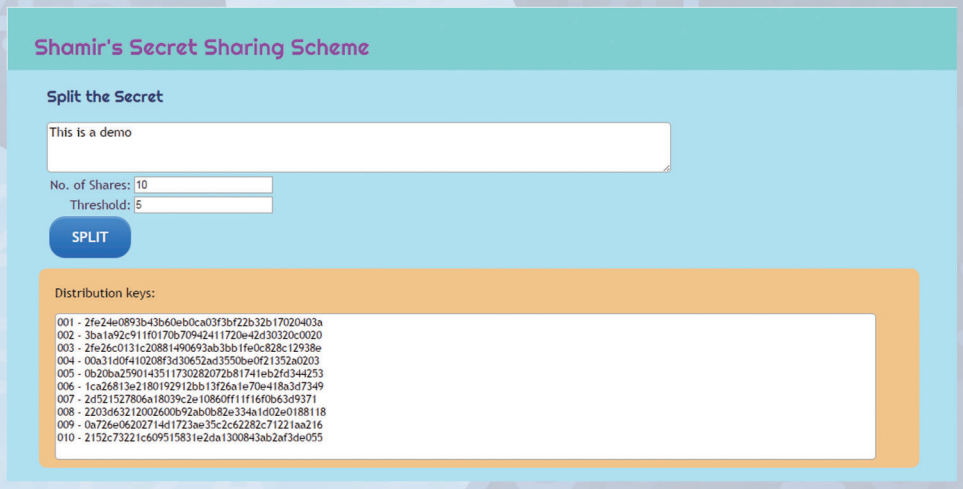
2. Creates an array to store the x and y values in decimal format of the corresponding elements from the input.

3. Creates an array called "poly" for integrating x and y values in "lines". In each "poly", a 2D array "shares" stores the corresponding x and y values by extracting from the keys.

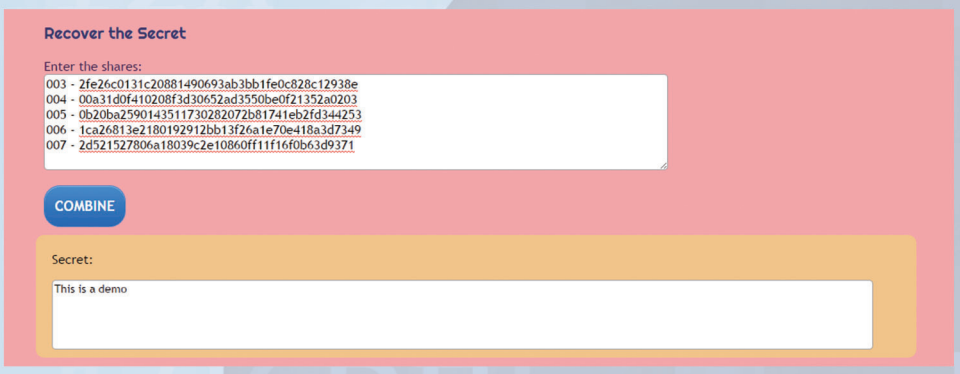
4. Calculates the initial ASCII value of that character in each "poly" and aggregates all these characters into a string.

Results

Graphic User Interface



This is the page of splitting the secret message to ten shares.



This is the page of recovering the secret from arbitrary five shares.

Evaluation

Accuracy: The program can correctly splits the message and reconstructs the message when there are sufficient shares.

Running time: It depends on the length of the secret message and number of thresholds. Generally, the running time is acceptable when both the length of the secret message and number of thresholds are in reasonable range. (Length of message: a paragraph number of threshold: within 50)

GUI: User-friendly layout with simple buttons.

Consistency: Same result occurs in different machines and different browsers for recovery

Mobility: Able to access the webpage with any electronic devices if internet connection is available.

Conclusion

The webpage can splits and recovers the secret message with no error now. However, if the algorithm is enhanced and the code is improved to be more concise, it is believable that the length of the message can be longer and number of thresholds can be larger.