How to share secret?

—A Theory About Cryptographic Sharing

Jinyang Huang 47355611
Xizheng Pan 45702163

Importance of password

—Safety

—Easy to save

Drawbacks:

—Full permission

—Loss

—>Sharing

—>Back up

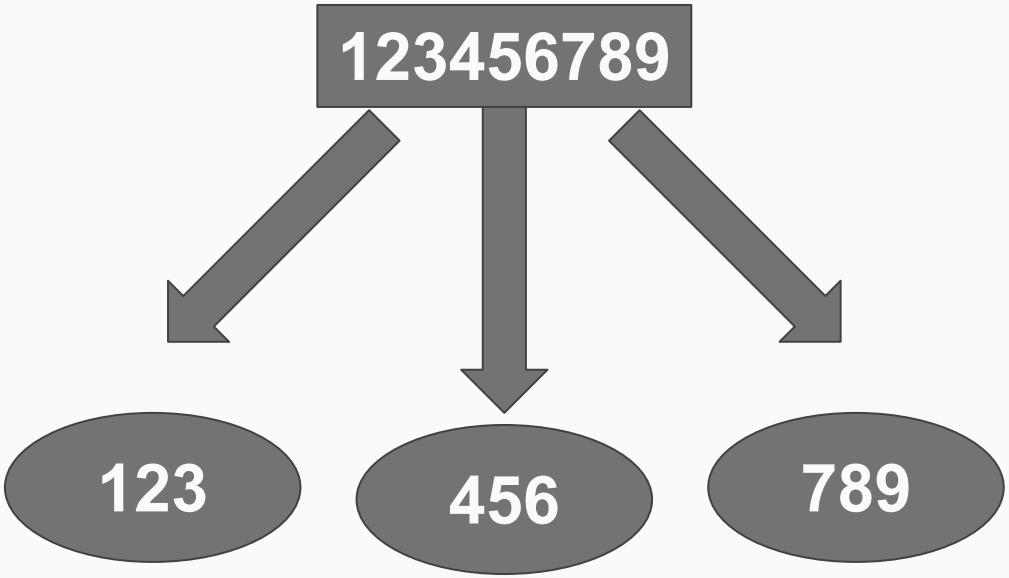
Secret sharing

—Divide password

—Password reconstructure

Potential issues:

–Easy to be cracked



Shamir

—Adi Shamir

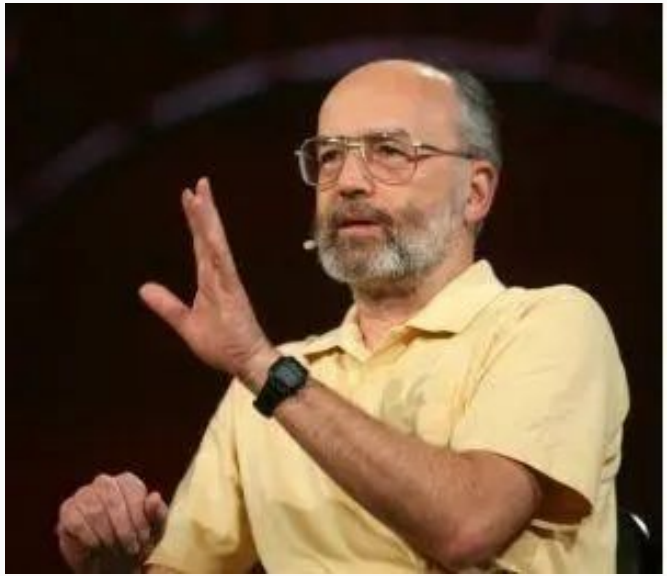
Expert in cryptography

—ASR

Asymmetric cryptographic algorithm

—1979

secret sharing



Method

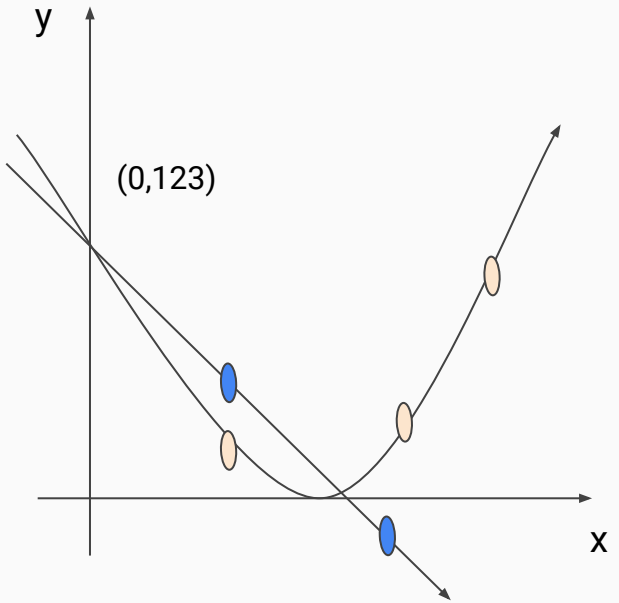
- Let's say **S** is the secret that we wish to encode.
- It is divided into N parts: S1, S2, S3, …., Sn.
- After dividing it, a number **K** is chosen by the user in order to decrypt the parts and find the original secret.
- It is chosen in such a way that if we know less than K parts, then we will not be able to find the secret S (i.e.) the secret S can not be reconstructed with (K – 1) parts or fewer.
- If we know **K** or more parts from S1, S2, S3, …., Sn, then we can compute/reconstructed our secret code S easily. This is conventionally called (K, N) threshold scheme.

Examples

S=65

N=4

K=2



Examples

$$l_i = \frac{x - x_0}{x_i - x_0} \times \ldots \times \frac{x - x_{i-1}}{x_i - x_{i-1}} \times \frac{x - x_{i+1}}{x_i - x_{i+1}} \times \ldots \times \frac{x - x_{k-1}}{x_i - x_{k-1}}$$

$$f(x) = \sum_{i=0}^{K-1} y_i l_i(x)$$

Answer

$$l_0 = \frac{x - x_1}{x_0 - x_1} = \frac{x - 3}{1 - 3}$$

$$l_1 = \frac{x - x_0}{x_1 - x_0} = \frac{x - 1}{3 - 1}$$

$$f(x) = y_0 l_0 + y_1 l_1$$

$$f(x) = 80\left(\frac{x - 3}{-2}\right) + 110\left(\frac{x - 1}{2}\right)$$

$$f(x) = -40x + 120 + 55x - 55$$

$$f(x) = 15x + 65$$

Advantages

(1) A reasonable backup is created for the key, which overcomes the disadvantage that the greater the number of saved copies, the greater the risk of security leakage, and the smaller the saved copy, the greater the risk of copy loss.

(2) It is beneficial to prevent the problem of excessive concentration of power leading to abuse.

(3) The attacker must obtain enough subkeys to recover the shared key, which ensures the security and integrity of the key.

(4) The reliability of the system is increased without increasing the risk.

G.R.Blakley’s threshold scheme

—Blakley proposed to use more than two-dimensional space to encrypt

define n,t

set key in t-dimensional space (e.g.(a,b,c))

construct n planes passing through this point：

x+y+z; x+y+2z; x+y+3z……

—Minimum t keys to solve t level secret.

Construction

—(m,t)

m: number of whole users

t: threshold number

@return m secret array

```java
public static BigInteger[] share(BigInteger secret, int m, int t) {
    //save t
    BigInteger[] coefficients = new BigInteger[t];
    coefficients[0] = secret;
    for (int i = 1; i < t; i++) {
        coefficients[i] = generateRandomBigInteger();
    }

    BigInteger[] userShares = new BigInteger[m];
    //process sharing
    for (int i = 0; i < m; i++) {
        userShares[i] = computeShare(coefficients, (i + 1));
    }

    return userShares;
}
```

Reconstruction

```
public static BigInteger reconstruction(BigInteger[] shares, int t) throws Exception {
    int n = shares.length;
    if (t > n) {
        throw new Exception("unreached secret");
    }
    BigInteger result = new BigInteger("0");
    for (int i = 0; i < t; i++) {
        result = result.add(interpolation(shares, i + 1, t));
    }

    return result.mod(p);
}
```

Lagrange polynomial

```
public static BigInteger interpolation(BigInteger[] values, int xK, int t) {
    BigInteger result;
    //calculate f(0)
    BigInteger zero = BigInteger.ZERO;
    BigInteger x_k = new BigInteger(String.valueOf(xK));
    //Lagrange polynomial
    BigInteger up = BigInteger.ONE;
    BigInteger down = BigInteger.ONE;
    //i: user i
    for (int i = 0; i < t; i++) {
        BigInteger x_i = new BigInteger(String.valueOf((i + 1)));
        if (x_i.equals(x_k)) {
            continue;
        }
        up = up.multiply(zero.subtract(x_i));
        down = down.multiply(x_k.subtract(x_i));
    }
    result = up.multiply(down.modInverse(p));
    result = result.multiply(values[xK - 1]);
    return result;
}
```

Development

—Data transfer and data recovery

error correction during storage or transfer

—The way to protect secrets does not necessarily depend on algorithms alone,

perhaps by adjusting the structure of personnel or the process of verifying keys

Conclusion

—Secret protection

—Shamir

—Methods

—Others’ researching

—Developing

Thanks ！

References

[1]Shamir, A. (n.d.). How to Share a Secret. https://dl.acm.org/doi/pdf/10.1145/359168.359176

[2]Shamsoshoara, A. (n.d.). OVERVIEW OF BLAKLEYS SECRET SHARING SCHEME Predictive Communication for UAV Networks View project OVERVIEW OF BLAKLEYS SECRET SHARING SCHEME. https://doi.org/10.13140/RG.2.2.10037.32488

[3]https://en.wikipedia.org/wiki/Shamir%27s_Secret_Sharing

[4]coding: https://blog.csdn.net/weixin_45071560/article/details/124482046