深圳大学考试答题纸

(以论文、报告等形式考核专用)

二〇 21 ～二〇 22 学年度第 2 学期

课程编号 1500210001R 课序号 1 课程名称 离散数学(荣誉) 主讲教师 闫巧 评分

学 号 2020151002 姓名 郑杨 专业年级 2020 级软件工程腾班

教师评语:

题目: Application of Number Theory in Public Key Cryptography

Application of Number Theory in Public Key Cryptography

Yang Zheng

June 1, 2022

Abstract

I reviews the historical development of cryptography, and expounds the basic idea of public key cryptography, a branch of modern cryptography. The classic algorithms of public key cryptography are reviewed, the RSA algorithm and ElGamal algorithm are described in detail, and the knowledge of number theory is used to prove their correctness and security.

1 Introduction

Cryptography was born as early as 400 BC, and its development can be roughly divided into three stages: the classical cryptography stage before 1949; the modern cryptography stage from 1949 to 1975, when cryptography gradually became a branch of science; After 1976, entering the stage of modern cryptography, symmetric key cryptography has been further developed, resulting in a new direction of cryptography - public key cryptography .The basic idea of public key cryptography was first proposed by R. Merkle in 1974[3], and then Diffie and Hellman first proposed the concept of "public-key cryptography" based on the one-way trapdoor function[1]. In this type of cipher, different keys are used for encryption and decryption. Among them, the one used for encryption is called the public key, and the one used for decryption is called the private key.

Before the first public key encryption algorithm was proposed, all cryptosystems were symmetric encryption systems, as shown in Figure 1, but there are many problems with this system, one of which is that since there is only one key for encryption and decryption, Both sender and receiver must know the key. That is to say, the ciphertext and the key need to be sent by the sender to the receiver at the same time, which also raises a question, how should the key be sent to ensure security? This problem is essentially the same as the encrypted information. The answer is to encrypt the key again, but this does not solve the problem at the root cause, because a new key will be generated after the key is encrypted, so as you can see Yes, the problem is an infinite loop of encryption and keys. The key agreement algorithm proposed by Diffie and Hellman in[1] has solved the leakage problem of the key delivery process. Then, there are still key management and unilateral leakage problems in the symmetric encryption system.

The new system, as shown in Figure 2, proposed by Diffie and Hellman in[1] solves the problem of key management and leakage. Their ciphers are asymmetric ciphers with two different keys: public key and private key. The private key is the decryption key and the public key is the encryption key. If a person Bob wants to send an encrypted message to another person Alice, all he needs to do is encrypt it with the public key released by Alice, then the only person in the world who can decrypt this ciphertext is Alice, because only she has the private key. Such encryption rules are based on the one-way trapdoor function, that is, it is almost impossible to calculate the private key through the public key, which ensures security. At the same time, users in each system only need to

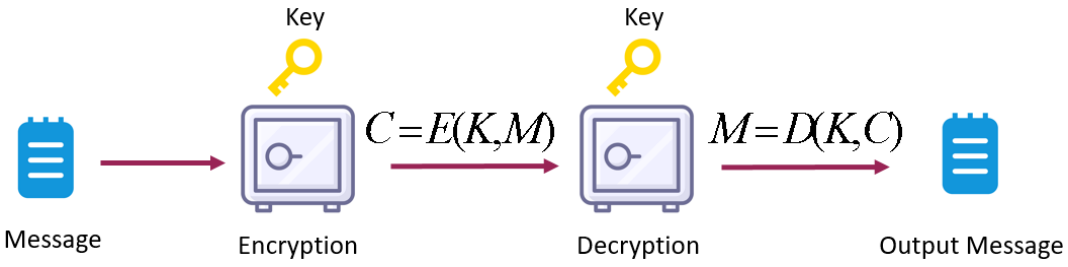

Figure 1: Symmetric Encryption System

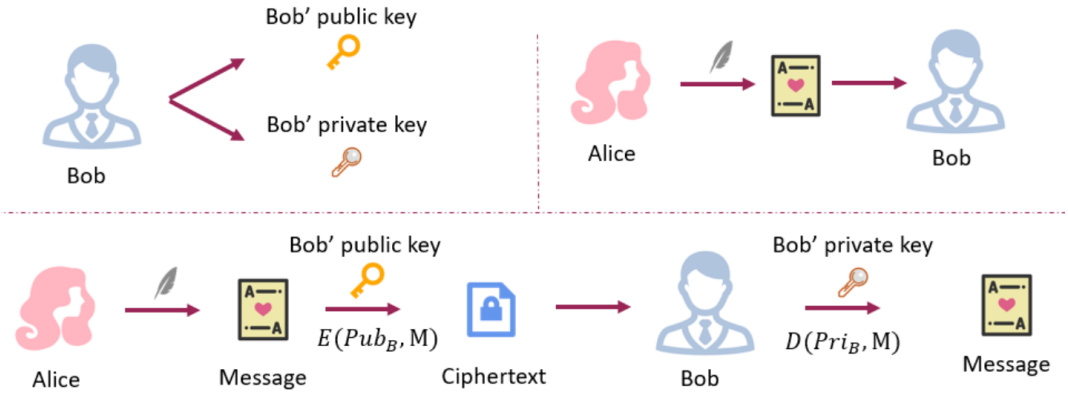

Figure 2: Public-Key Cryptosystem

manage their own private keys, reducing the burden of managing keys. However, Diffie and Hellman just provided the basic concept, and did not find a one-way trapdoor function that meets their requirements.

The team of Rivest, Shamir and Adleman, based on the ideas of Diffie and Hellman, finally found a function that meets the requirements after a year, and proposed the corresponding encryption and decryption algorithm[4]. In addition to encryption and decryption functions, this algorithm also has electronic signature functions. Information can be "signed" using a privately held decryption key, and this signature can be verified by anyone using the pair's public key. The signature cannot be forged, and the signer cannot deny the validity of the signature. The algorithm they proposed is called the RSA algorithm.

After the RSA algorithm was proposed, public key encryption algorithms such as ElGamal[2] and elliptic curves were successively proposed, and cryptography entered a new development period. Generally speaking, the security of public key cryptography is guaranteed by the intractability of the corresponding mathematical problems on the computer, and some related knowledge of number theory can be used to explain its security. This article introduces the RSA and ElGamal algorithms in detail.

2 Public-Key Cryptosystems

In the Public-Key Cryptosystems, the encryption algorithm is generally public, and it is recommended to make an encryption algorithm public to stand the test of time and users. Therefore, under the premise that the encryption algorithm is disclosed, the security of encryption depends on the security of the key.

The Public-Key Cryptosystems regard the encryption and decryption process as two kinds of procedures, which are represented by E and D respectively, and the plaintext message and the ciphertext message are represented by M and C respectively, then the Public-Key Cryptosystems has the following four characteristics:

1. Deciphering the enciphered form of a message M yields M. Formally,

$$D(E(M)) = M \tag{1}$$

2. Both E and D are easy to compute.

3. By publicly revealing E the user does not reveal an easy way to compute D. This means that in practice only he can decrypt messages encrypted with E, or compute D efficiently.

4. If a message M is first deciphered and then enciphered, M is the result. Formally,

$$E(D(M)) = M \tag{2}$$

Among them, the property 1 and 2 are also established in the traditional symmetric encryption system, which ensures the original purpose of encryption. Property 3 guarantees the security of the key, that is, the private key cannot be easily calculated using the public key. Property 4 enables the signature function.

Each user in the public key system will generate his own public key and private key. The public key is publicly stored in public files for other users to view, and the private key is kept by himself.

3 RSA

RSA is a method for obtaining digital signatures and Public-Key Cryptosystems proposed in [4]. In this section, I review the key-generation methods of RSA, the encryption and decryption methods of RSA, digital signatures using RSA and proof the correctness and safety of them.

3.1 Key Generation

The process of generating public key(e, n) and private keys(d, n) for each user in the system is as follows:

1. Select two random primes (very large prime) p,q.
2. Compute n as the product of p and q: $n = p * q$.
3. Select a large, random integer d which is relatively prime to $\varphi(n)$.Formally,
$$gcd(d, (p-1)(q-1)) = 1 \tag{3}$$
4. The integer e is finally computed from p, q and d to be the "multiplicative inverse" of d, module $(p-1)(q-1)$.Formally,
$$ed \equiv 1(mod\ (p-1)(q-1)) \tag{4}$$

As can be seen from the above key generation process, the following information will be generated:

$$p, q, n, \varphi(n), d, e$$

In addition to e and n being public, other information needs to be strictly confidential, the reason will be mentioned in Section 3.5. The next section will introduce the RAS algorithm encryption and decryption methods.

3.2 Encryption and Decryption Methods

Assuming that Alice and Bob are two users in the Public-Key Cryptosystems, Alice wants to send a piece of encrypted information C to Bob, and Alice has obtained Bob's public key (e, n). The encryption and decryption process(as shown in Figure 3) is as follows:

1. Alice decomposes the plaintext into several blocks, so that each block can be represented as an integer of $[1, n-1]$. Since blocks are independent of each other, only a certain block is considered during encryption, assuming the information representation of a certain block is an integer M.
2. Alice uses Bob's public key to perform the following operations to obtain encrypted information C, and sends C to Bob.
$$C = M^e\ mod\ n \tag{5}$$
3. After Bob receives C, he uses his reserved private key (d, n) to perform the following operations to decrypt, and obtain M.
$$M = C^d\ mod\ n \tag{6}$$

The next problem is to prove the correctness of the two formulas (5) and (6), which will be explained in the next section.

3.3 Mathematical Principles

We use Euler's theorem and Fermat's little theorem for auxiliary proofs.

Theorem 3.1 *For any integer M which is relatively prime to n,*

$$M^{\varphi(n)} \equiv 1\ (mod\ n) \tag{7}$$

Theorem 3.2 *if integer M is relatively prime to prime integer p, then we have,*

$$M^{p-1} \equiv 1\ (mod\ p) \tag{8}$$

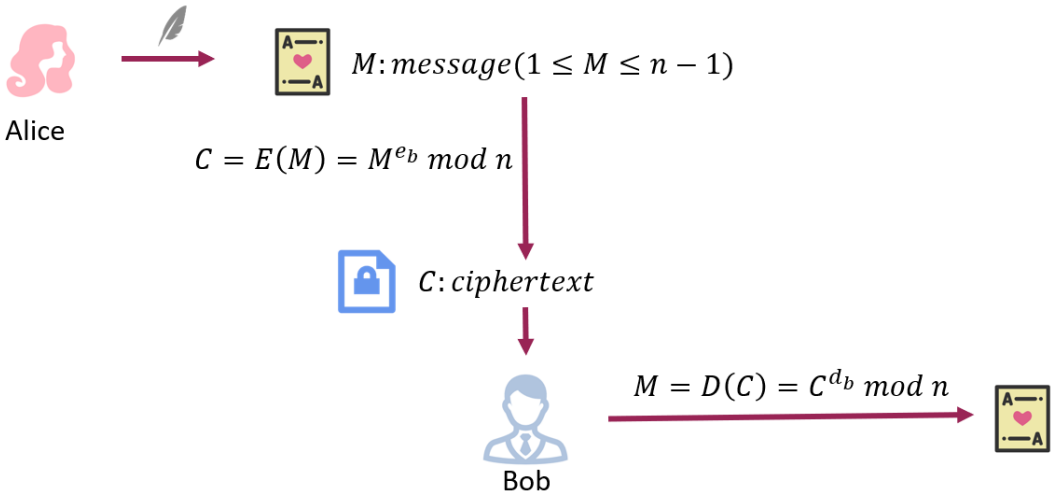

Figure 3: Encryption and Decryption Methods

Here $\varphi(n)$ is the Euler totient function giving the number of positive integers less than n which are relatively prime to n. For prime number p, we have

$$\varphi(p) = p - 1 \tag{9}$$

Moreover, Euler totient function is an integral function, which means if the integer a is relatively prime to other integer b,

$$\varphi(ab) = \varphi(a)\varphi(b) \tag{10}$$

In our case, we have

$$\varphi(n) = \varphi(pq) = \varphi(p)\varphi(q) \tag{11}$$

Proof 3.1 *Now, we want to prove the following two formulas,*

$$M \equiv D(E(M)) \equiv (E(M))^d \equiv (M^e)^d \equiv M^{ed} \ (mod\ n) \tag{12}$$

$$M \equiv E(D(M)) \equiv (D(M))^e \equiv (D^d)^e \equiv M^{ed} \ (mod\ n) \tag{13}$$

According to the formula(4)*, that is equal to proving the following formula,*

$$M^{k\varphi(n)+1} \equiv M(mod\ n) \tag{14}$$

It can be divided into the following two cases to prove the equation(13)*.*
case 1. $gcd(M, n) = 1$*. According to theorem* 3.1*, we can easily get,*

$$(M^{\varphi(n)})^k M \equiv M(mod\ n) \iff M^{k\varphi(n)+1} \equiv M(mod\ n) \tag{15}$$

case 2. $gcd(M, n) \neq 1$*. We set* $M = tq$*, thus we have* $gcd(M, p) = 1$*. According to theorem* 3.2*, we have*

$$M^{k\varphi(n)+1} \equiv M(mod\ p) \tag{16}$$

so, we have

$$(tq)^{k\varphi(n)+1} = yn + tq \iff M^{k\varphi(n)+1} \equiv M(mod\ n) \tag{17}$$

3.4 Details

In this section, I summarize some details in the RSA algorithm, they include how to encrypt and decrypt efficiently, how to find large prime numbers, how to choose d and how to compute e from d and $\varphi(n)$.

3.4.1 How to encrypt and decrypt efficiently

In order to encrypt and decrypt the message M, we only need to calculate the formula (5) and (6).

For example, to compute $M^e \ mod\ n$ efficiently, we can use an algorithm called "Fast Exponentiation Algorithm".Its pseudo code is as follows,

Algorithm 1 Fast Exponentiation

Input: M, e, n
Output: $M^e \ mod \ n$

```
function Fastpow(M, e, n)
    res ← 1
    while e ≠ 0 do
        if e & 1 ≠ 0 then
            res ← res * M(mod p)
        end if
        M ← M * M
        e ← e >> 1
    end while
    return res
end function
```

3.4.2 How to find large prime numbers

During key generation, two large primes(p and q) need to be found, and in Section 3.6 I will explain why the decimal length of the primes needs to be very long (hundreds of digits).

Theorem 3.3 *if integer M is relatively prime to prime integer p, then we have,*

$$M^{p-1} \equiv 1 \ (mod \ p) \tag{18}$$

According to the prime number theorem, to find the first prime number less than n, only about $\frac{ln(n)}{2}$ prime number checks are required. For example, to find a 2048-bit prime number, you can detect the primeness of $\frac{ln2^{2048}}{2} \approx 710$ randomly selected 2048-bit integers. There are some primality detection algorithms that we can use, the Fermat test is one of them, but it's not quite right (for example, $7^{560} \equiv 1(mod \ 561)$, but 561 is a composite number, $561 = 3 * 11 * 17$, and something like 341, 645, 1105, etc., the numbers that are misjudged in this way are called Carmichael numbers). The Miller-Rabin test will be more accurate. In fact, testing large prime numbers is a probabilistic job.

3.4.3 how to choose d

It is very easy to choose an integer d which satisfies formula(3). Obviously, all integers greater than $max(p, q)$ can satisfy this requirement

3.4.4 how to compute e from d and $\varphi(n)$

It can be seen from the formula(4) that e is the multiplicative inverse of d with respect to modulo $\varphi(n)$. This problem can be solved quickly using the Extended Euclidean Algorithm.

3.5 Security

The security of an encryption algorithm is usually proven by negative cracking methods. In the RSA encryption algorithm, only the public key is disclosed, so the cracker can only use these two information to crack the private key.

The usual cracking method is as follows,

1. Factoring n
2. Computing $\varphi(n)$ without factoring n
3. Determining d without factoring n or computing $\varphi(n)$
4. Computing D in some other way

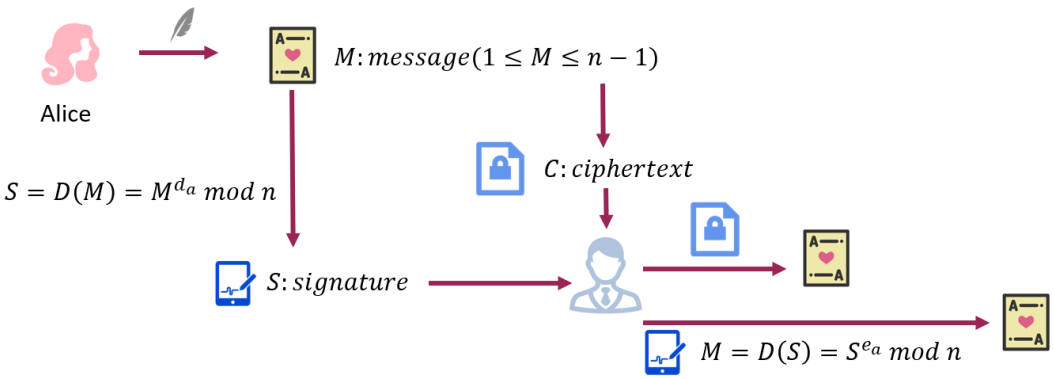

Figure 4: Digital Signature with RSA

The complexity of these methods has been proved to be no less than the complexity of prime factorization n, and there is no very efficient algorithm at present. In theory, as long as the key space is limited, it can be cracked with a lot of computation and a lot of time. However, as long as the key length is selected enough, it becomes impossible to crack the algorithm in real time, so the security of RSA is actually It's discussing a timeliness issue. With the upgrade of hardware computing speed, the number of key bits that was considered to be sufficiently secure in the past will be gradually broken, and the length of the key has to be further increased. Therefore, the key length of the RSA algorithm is generally longer than that of symmetric encryption and elliptic curve encryption. longer in length.

3.6 Signature

The signature algorithm is generally used to verify the identity of the information sender. In a public key encryption system, the sender can use its own private key to generate a message signature, and the receiver can successfully verify the signature only by using the sender's public key. This ensures that the signature cannot be forged, and it also makes it impossible for the sender to deny its own signature.The flow chart of using the RSA signature algorithm is shown in Figure 4.

4 ElGamal

In [1], Diffie and Hellman proposed a key distribution algorithm based on discrete logarithm. Inspired by their ideas, ElGamal then proposed a new public key encryption algorithm, which security depends on the difficulty of computing discrete logarithms finite fields.

In this section, I first review the key agreement algorithm proposed by Diffie and Hellman in [1], and then focused on the public key encryption algorithm proposed by ElGamal in [2] and the application of number theory in proving its correctness and security.

4.1 Review of Diffie-Hellman Key Distribution

Diffie-Hellman Key Distribution algorithm mainly solves the problem of key distribution, and is not used for encryption itself.

Suppose Alice and Bob need to negotiate a key, they need to go through the following process(as shown in Figure 5),

1. First, Alice and Bob share a prime number p and the generator $g(2 \leq g \leq p-2)$ of the prime number p.
2. Then Alice generates a random integer $a(1 \leq a \leq p-1)$, computes $y_a = g^a \ mod \ p$ and sends it to Bob; At the same time, Bob also generates a random integer $b(1 \leq b \leq p-1)$, computes $y_b = g^b \ mod \ p$ and sends it to Alice. Among them, a is private to Alice, b is private to Bob, and other information is shared.
3. Finally, Alice obtains the secret key K_a by calculating formula (19); Bob obtains the secret key K_b by calculating formula (20). It can be prove that K_a is equal to K_b.

$$K_a = (y_b)^a \ mod \ p \tag{19}$$

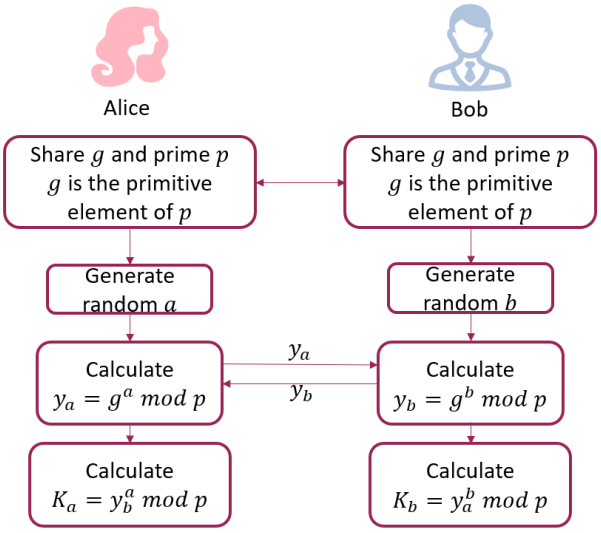

Figure 5: Diffie-Hellman Key Distribution

$$K_b = (y_a)^b \ mod \ p \tag{20}$$

Proof 4.1 *It is easy to proof K_a is equal to K_b as follow,*

$$K_a = (y_b)^a \ mod \ p = (g^b)^a \ mod \ p = g^{ab} \ mod \ p \tag{21}$$

$$K_b = (y_a)^b \ mod \ p = (g^a)^b \ mod \ p = g^{ab} \ mod \ p \tag{22}$$

Then if the eavesdropper Eve wants to crack the secret key, obviously the only information that Eve can eavesdrop on is p, g, y_a, y_b, but to calculate K_a or K_b, you need to know a or b.Taking calculating a as an example, can Eve calculate a according to the condition $g^a \ mod \ p = y_a$? This is actually quite difficult when p is a large prime number, which is the discrete logarithm problem.

4.2 Key Generation

In ElGamal's Public-Key Cryptosystem, the method of key generation is very similar to Diffie-Hellman key distribution algorithmm, where (g, p, y) is the public key and x is the private key. p must be chosen such that $p-1$ has at least one large prime factor, if $p-1$ has only small prime factors, then computing discrete logarithms is easy. g is a generator of prime number p. $x(1 \leq x \leq p-2)$ is a random integer user chooses and $y = g^x \ mod \ p$.

4.3 Encryption and Decryption Methods

Now suppose that Alice wants to send Bob a message M, where $0 \leq M \leq p-1$. The encryption and decryption methods(as shown in Figure 6) is as follows,

1. Alice select a random integer k.
2. Alice gets Bob public key y_b and compute $K = y_b^k \ mod \ p$.
3. Computes $C_1 = g^k \ mod \ p$ and $C_2 = K * M \ mod \ p$ and sends (C_1, C_2) as ciphertext to Bob.
4. When Bob receives the ciphertext (C_1, C_2), he can compute K by $K = C_1^{x_b} \ mod \ p$ and M by $M = C_2 * K^{-1} \ mod \ p$.

The correctness of ElGamal's encryption and decryption methods is obvious.Note that the size of the ciphertext is double the size of the message.

Breaking the system that ElGamal build is equivalent to breaking the Diffie-Hellman distribution scheme. Computing k or x_b from C_1, C_2 and y_b is equivalent to computing discrete logarithms. The reason is that both k and x_b appear in the exponent in y_b and C_1.

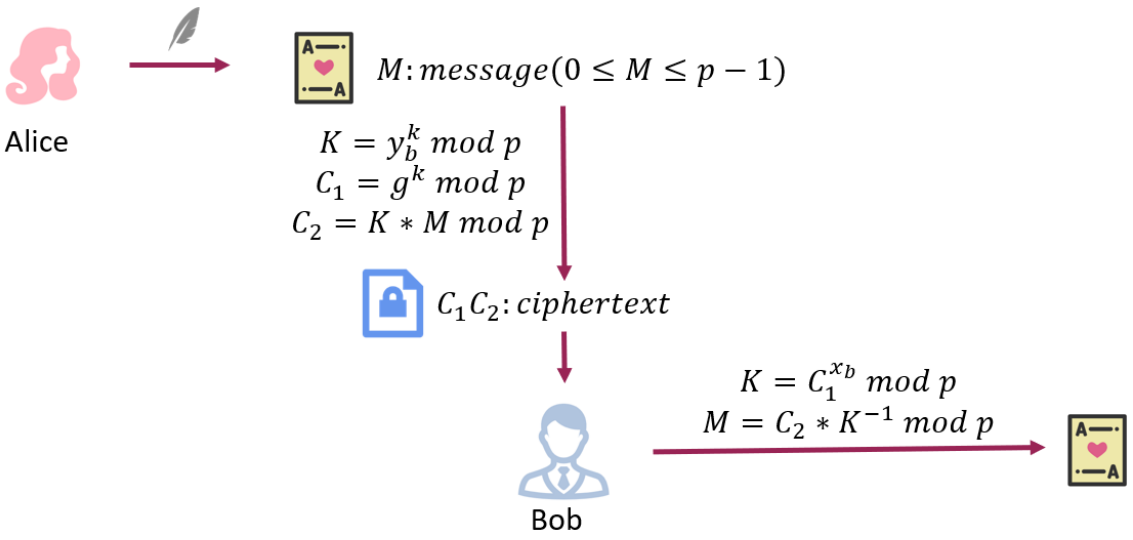

Figure 6: Encryption and Decryption Methods of ElGamal

4.4 Signature

A signature scheme proposed by ElGamal is described in this section. Let $M(0 \leq M \leq p-1)$ be a document to be signed, Alice be the signer. Alice first choose a random integer k, and with x_a(Alice's private key) and k, the signature text(r, s) can be computed as follows,

$$r = g^k \ mod \ p \tag{23}$$

$$s = (M - x_a^r) * k^{-1} \ mod \ (p-1) \tag{24}$$

When Bob receive the document M and the signature text (r, s), he can verify the identity of the signer by using Alice's public key y_a to test if the equation (25) is true.

$$y_a^r r^s \equiv g^M \ mod \ p \tag{25}$$

Proof 4.2 *Because only Alice has the private key x_a, no one but Alice can generate s. According to formula* (24)*, we get*

$$ks = (M - x_a r) \ mod \ (p-1) \iff ks = (M - x_a r) \ + h(p-1) \tag{26}$$

so, we have

$$g^{ks} = g^{(M - x_a r)} g^{h(p-1)} \tag{27}$$

Since g is a generator of prime number p, so we have

$$g^{ks} \equiv g^{(M-x_a r)} (mod \ p) \iff g^{ks} g^{x_a r} \equiv g^M (mod \ p) \tag{28}$$

According to formula (23) *and $y_a = g^{x_a} \ mod \ p$,*

$$r^s y_a^r \equiv g^M \ mod \ p \tag{29}$$

So, verifying the correctness of equation (24) *is equivalent to verifying that of equation* (25)*.*

5 Conclusion

This paper reviews the history of public key cryptography and introduces two classic public key cryptography algorithms based on different mathematical problems in detail, and explains the application of number theory knowledge in them. For security reasons, the length of the key generated by the public key encryption algorithm is generally longer and the encryption time is longer. In practice, it is often combined with the symmetric encryption algorithm, such as using the public key encryption algorithm to encrypt the key generated by the symmetric encryption algorithm.

References

[1] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976.

[2] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4):469–472, 1985.

[3] Ralph C Merkle. Secure communications over insecure channels. *Communications of the ACM*, 21(4):294–299, 1978.

[4] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.