A Brief Note on Cryptography

Ganyuan Cao[1] [2]

[1]Arizona State University
[2]gansama.github.io

Contents

Preface

A Brief Note on Cryptography is a brief explanation of theories of cryptography. This note is divided into five parts: Symmetric Cryptography, Asymmetric Cryptography, Applied Cryptography, Advanced topics in Cryptography, and Appendices.

In the Symmetric Cryptography part, some modern symmetric cryptosystems including AES are discussed. Some cryptographic hash functions and pseudorandom integer generators are also discussed in this part. In the Asymmetric Cryptography part, some asymmetric cryptosystems are discussed including RSA, ElGamal, and ECC. In Applied Cryptography, the ways Cryptography is applied to the network and real-life scenarios are discussed. In advanced topics in Cryptography, we discussed some topics which are recently discussed in the area of Cryptography including Zero-knowledge Proof, Multip-party computation, etc. In Appendices, mathematical theories are discussed from the areas of number theory, abstract algebra, and elliptic curves.

Modern Cryptography has four main objectives which are confidentiality, data integrity, authentication, and reputation. This note also discussed how different cryptography primitives utilize those four objects.

Cryptography is strongly connected with mathematics theories such as Abstract Algebra, Number Theory, etc. Since those mathematics theories require knowledge from discrete mathematics and linear algebra, it is assumed that the reader has that knowledge.

I also provided some codes which are related to the content of this note at my github repository https://github.com/GanSama/Crypto_Func.These codes are written in *Python*, *Pari/GP* and *Wolfram Mathematica*. If you would like to view those codes, please clone the repository and install *Wolfram Mathematica* since the codes for *Wolfram Mathematica* can not be viewed in a text editor.

About author's website

- `gansama.github.io` is the personal website of the author.
- `https://github.com/GanSama/Crypto_Func` is the github repository which includes the code for reference.

Acknowledgements

A special word of thanks goes to Dr Andrew Bremner[1], Dr John Jones [2] ,and Dr Nancy Childress [3] for the explanation on theories of mathematics.

[1] `https://math.la.asu.edu/~andrew/`
[2] `https://hobbes.la.asu.edu/`
[3] `https://math.la.asu.edu/~nc/`

Chapter 1

Introduction and Classical Ciphers

1.1 Ciphers and Cryptosystems

In Cryptography, a cipher and a cryptosystem share a similiar concepts. A cipher is an algorithm for performing encryption or decryption while a cryptosystem is a suite of cryptographic algorithms which are implemented for specific scenarios.

We define A cryptosystem as a 5-tuple $(P, C, K, \epsilon, \delta)$ where

- P is plaintext space i.e., the set of plaintexts.
- C is ciphertext space i.e., the set of ciphertexts.
- K is key space i.e., the set of keys.
- $\epsilon = \{E_k | k \in K\}$ where E_k is an encryption function using key k.
- $\delta = \{D_k | k \in K\}$ where D_k is a decryption function using key k.

The encryption function is defined as:

$$E_k : P \times K \to C$$

The decryption function is defined as:

$$D_k : C \times K \to P$$

A cryptosystem contains at least 3 algorithms: KeyGen, Enc, Dec. Specifically,

- KeyGen generates keys (or key pair) in key space which are used to encrypt or decrypt.
- Enc utilizes encryption function E_k defined above.
- Dec utilizes decryption function D_k defined above.

In the following content of this note, a cryptosystem will be partitioned into those 3 parts and explained in details.

1.1.1 Kerckhoffs's principle

Kerckhoffs's principle was stated by cryptographer Auguste Kerckhoffs in the 19th century. The principle states: "A cryptosystem should be secure even if everything about the system, except the key, is public knowledge." Claude Shannon reformulated the principle as Shannon's maxim: "the enemry knows the system i.e., one ought to design systems under the assumption that the enemy will immediately gain full familiarity with them".

Contrary to Kerckhoffs's principle, security through obscurity keeps the encryption and decryption functions secret. Security through obscurity is considered vulnerable since the encryption/decryption functions can be derived via reverse engineering.

1.1.2 Confusion and Diffusion

Confusion and *Diffusion* are two important methods used for frustrating statistical analysis such that a secure cipher can be created. The way how these two methods are implemented in a cipher is important when evaluating the security of a cipher, especially symmetric encryption algorithms, hash functions, and pseudorandom number generators.

These two methods were first identified by Shannon in *A Mathematical Theory of Cryptography*. According to Shannon's definition, confusion refers to making the relationship between the ciphertext and key as complex and involved as possible. Diffusion refers to dissipating the statistical structure of plaintext into a statistical structure involving long combinations of letters in the ciphertext. In simple words, confusion is used to creating uninformed ciphertext while diffusion is used to increase the redundancy of plaintext over ciphertext. In designing of a cipher, confusion means the method of substitution (i.e., $a \mapsto B$) while diffusion means the method of transposition or permutation (i.e., $abcd \mapsto DACB$)

1.2 Perfect Secrecy

In general, we say an encryption scheme has perfect secrecy if the ciphertext does not reveal any information of the plaintext. The formal definition of Shannon is:

Definition 1.2.1. An encryption scheme E_k with $|K| = |P| = |C|$ where K is the key space, P is plaintext space, and C is ciphertext space has perfect secrecy if and only if

1. Each key is used with the probability of $\frac{1}{|K|}$.
2. $\forall p \in P, \forall c \in C$, $\exists! k \in K$ s.t. $E_k(p) = c$.

1.3 Classical Ciphers

In this section, I will talk about some classical ciphers. In classical ciphers, we construct a 1-1 mapping between the alphabet and integer 0 to 25.

0	1	2	3	4	5	6	7	8	9	10	11	12
A	B	C	D	E	F	G	H	I	J	K	L	M
13	14	15	16	17	18	19	20	21	22	23	24	25
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

1.3.1 Shift Cipher

The Shift Cipher is also called the Caeser Cipher which was designed by Julius Caesar. We define the encryption function of Shift Cipher $E_k(m)$ where $k \in \mathbb{Z}^+$ is the key and m is a character of the plaintext.

$$E_k(m) \equiv m + k \pmod{26}$$

Shift Cipher is simple shifting of character on alphabet. To decrypt the ciphertexts, the receiver could just shift back. We define the dncryption function of Shift Cipher $D_k(c)$ where k is the key and c is a character of the ciphertext

$$D_k(c) \equiv c - k \pmod{26}$$

To break the Shift Cipher, an attack could use frequency analysis or the attack could just use brute-force to break the cipher by trying all 26 combinations.

1.3.2 Affine Cipher

Compared with Shift Cipher, Affine Cipher provides a better security by utilizing a linear combination. The key of an affine cipher is a tuple $(\alpha, \beta) \in \mathbb{Z}^+ \times \mathbb{Z}^+$. We define the encryption function of Affine Cipher $E(m)$ where m is a character of the plaintext.

$$E(m) \equiv \alpha \cdot m + \beta \pmod{26}$$

The choice of α should satisfy the condition that $gcd(\alpha, 26) = 1$ to ensure α has a multiplicative inverse mod 26. If not, the ciphertexts could not be decrypted. Now we define the decryption function of Affine Cipher $D(c)$ where c is a character of the plaintext.

$$D(c) \equiv \alpha^{-1}(c - \beta) \pmod{26}$$

Similar to Shift Cipher, an attacker could try to break an affine cipher by frequency analysis. Compared with Shift Cipher, an attacker must try $12 \cdot 26 = 312$ if the attacker want to brute-force the cipher since there are 12 choices of α and 26 choices of β.

1.3.3 Vigenère Cipher

Vigenère Cipher is built on the top of Shift Cipher. The key of a Vigenère Cipher is a vector $(k_1, k_2, \cdots, k_n)$ where $k_i \in \mathbb{Z}$ for all $i = 1, 2, \cdots, n$. For each group of n characters, apply encryption with k_i on the i_{th} character. Let M be the message to be encrypted and $m_1, m_2, \cdots, m_l$ be characters in M. Let $(k_1, k_2, \cdots, k_n)$ be the encryption key. Assume that $l \geq n$, then the encryption function of Vigenère Cipher is defined as

$$E_{k_i}(m_j) \equiv m_j + k_i \pmod{26} \text{ where } j \equiv i \pmod{n}$$

The security of Vigenère Cipher depends on the key length. With longer key, the securer Vigenère Cipher is.

Attack on Vigenère Cipher
The first step of an attack on Vigenère Cipher is to determine the key length. The idea of the method to determine key length is from Index Coincidence (IC). Write two copies of a Vigenère Cipher ciphertexts on two long tapes. Shift one tape and find the offset which produce the greatest coincidence. Then the length of offset is likely to be the key length. With the longer ciphertext, it is easier to derive the key length. I will illustrate an example below.

Example 1.3.1. Let the ciphertext be

```
UYIBWEGUSSIRGAQOMLFUEFMEFNQJRRAXDVXYRLRPVUJYDFLVFUUEAZGB
TJWLFOMMGCNLUUCKUUFJJKUYIPVUZYMOXRASFIMETBQXEJAIFBKVACGT
EKNFXBXKUYNFWKUYEBMUUYIBWWBLMTLFENFJQVGBQGMWGBNFWKCODFQR
GBQNEKVWUBRZANTFAFEFPTMEPYTJWTUUDBGKRLIBWRFVQBYKVZGMEEQW
MOHZQUEIMJZCZELVNFIBCJZUPFXYRJAJRKGBMULZFZDJIEQUZEGFYFMC
SINNASPZGNXFAFBXIBWRAUBQVVPCMCPPZIDFTFJYDGYCZUFIIDNNUDMR
ANTBRYRQMTEEQNTBXYVMBSSKRAQSEDNHGKEEEYMMPPUUPOEKHLMMKVAC
GTMEGBQTIEFYFISLTBZPXKBNTFIOGYZUEEQHAULZAAXJOVFIQGJVPNUW
          ICLNTBXKUYSSIRGYEUQRGBQNEKVWUBRJUUPJX
```

Shift two indices, we have 28 coincidence

```
U Y I B W E G U S S I R G A Q O M L F U E F M ···
    U Y I B W E G U S S I R G A Q O M L F U E F M ···
```

Shift three indices, we have 14 coincidence

```
U Y I B W E G U S S I R G A Q O M L F U E F M ···
    U Y I B W E G U S S I R G A Q O M L F U E F M ···
```

Continue shifting, we have 33 coincidences when 6 indices are shifted. Then it is likely that 6 is the key length.

After we obtained the key length, we can try to derive the key with frequency analysis. Put the ciphertext into a matrix which has 6 columns and count the frequency of the first column :

Consider the possibilities that e appears at the first index of the plaintext is lower than the letter h,s,i and t. Using frequency analysis, it can be seen that U appeared most in the ciphertext, which is 9 times. By assuming the key is 13, G is decrypted to r which is also reasonable according to the frequency table. Thus, the first key may be N

Then, consider the second column, by counting frequency, it can be seen that U is most frequent. Since Y, which is second most frequent, is decrypted to i with this key which also makes sense. Thus, it can be assumed that the second key is 16, which is Q. Also, the letter Y which appeared second most frequently may also be the key, assume that Y is decrypted to e, thus the potential key may be 24, which is U.

Similarly with the third column, the letter appeared in the third column most is Q. Thus, the key may be 12, which is M.

Consider the fourth column, the letter appeared most is B. One potential key is 24, which is Y. Also, consider the second most frequent letter, which is F. Thus, the key may be 1, which is B.

Vigenère cipher usually uses a word as the key. To be English-text word, that the most possible key vector is (N, U, M, B, key_5, key_6). Thus, the potential keyword may be (N,U,M,B,E,R). Use the keyword to decrypt the ciphertext.

1.3.4 Playfair Cipher

1.3.5 Hill Cipher

Hill Cipher is a block cipher. The key of a Hill Cipher is a $n \times n$ matrix whose determinant is relatively prime to 26. To encrypt a message, divide the message into vector of length n and compute matrix multiplication.

Example 1.3.2. Let `abc` be the message to be encrypted. Then write the message as $(0, 1, 2)$. Let K be a matrix modulo 26.

$$K = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 11 & 9 & 8 \end{pmatrix}$$

Compute

$$(0, 1, 2) \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \equiv (0, 23, 22) \pmod{26}$$

to obtain the ciphertext `AXW`.

To decrypt the ciphertext `AXW`, we need to compute a inverse matrix modulo 26 of the encryption matrix K. Consider that $det(K) = -3$. Thus we have the inverse matrix as :

$$K^{-1} = -\frac{1}{3} \begin{pmatrix} -14 & 11 & -3 \\ 34 & -25 & 6 \\ -19 & 13 & -3 \end{pmatrix}$$

Also, $17 \cdot -3 \equiv 1 \pmod{26}$. Thus we have that

$$K^{-1} \equiv \begin{pmatrix} 22 & 5 & 1 \\ 6 & 17 & 24 \\ 15 & 13 & 1 \end{pmatrix} \pmod{26}$$

Then compute

$$(0, 23, 22) \begin{pmatrix} 22 & 5 & 1 \\ 6 & 17 & 24 \\ 15 & 13 & 1 \end{pmatrix} \equiv (0, 1, 2) \pmod{26}$$

to obtain the plaintext $(0, 1, 2)$

1.4 Classification of Modern Cryptography

In modern cryptography, there are two types: symmetric cryptosystems and asymmetric cryptosystems.

In symmetric cryptosystems, the encryption and decryption process are done using the same key i.e.,

$$\exists k \in K, \exists p \in P, \exists c \in C, E_k(p) = c \text{ and } D_k(c) = p$$

While in asymmetric cryptosystems, there are public key and private key. Public key is visable to all participants and it is used for encryption. Private key is used for decryption and digitial signature and it is kept secret for the owner of the key only. i.e.,

$$\exists priv, pub \in K, \exists p \in P, \exists c \in C, E_{pub}(p) = c \text{ and } D_{priv}(c) = p$$

With the *priv* and *pub* above,

$$\exists p \in P, \exists sig \in C \text{ s.t. } E_{priv}(p) = sig \text{ and } D_{pub}(sig) = p$$

The appearence of asymmetric cryptosystems enables the secure communication in long distance without a shared secret key. The possibility that the secret key is compromised is

elimated. However, since asymmetric cryptosystems usually require the modular operations on integers. It should not be used to encrypt data of large size since it is slow for a computer to do that. On the contrary, symmetric cryptosystems usually do bitwise operations with which the computers can process faster. Thus asymmetric cryptosystems are usually used to transmit the secret key for symmetric cryptosystems. After a secure communication channel is established, the file of large size can be encrypted using symmetric cryptosystems.

Part I

Symmetric Cryptography

Chapter 2

Advanced Encryption Standard : Rijindael

AES is a Rijndael block cipher. The standard was established by NIST in 2001. AES uses block size of 128 bits. The key length of AES includes 128, 192 and 256 bits. AES utilizes operations on finite field to provide security. There are four steps of an AES encryption.

2.1 Encryption Scheme

2.2 Layers

2.2.1 The ByteSub Transformation

2.2.2 The ShiftRow Transformation

2.2.3 The MixColumn Transformation

2.2.4 AddRoundKey

2.3 Decryption Scheme

2.4 Mode of Operation

2.4.1 Cipher Block Chaining (CBC)

2.4.2 Electronic Codebook (ECB)

Chapter 3

Message Authentication Code (MAC)

Chapter 4

Cryptographic Hash Function

Cryptographic Hash Function provides the way to verify the data integrity of a message. A hash function takes a message m of any length and outputs a digest of fixed length. Cryptographic hash functions are designed to be computationally infeasible and collision-resistance. Let H be the cryptographic hash function. It is computationally infeasible to compute message m from $H(m)$. To say that H is collision-resistance, let m_1, m_2 be message, then it is nearly impossible that $H(m_1) = H(m_2)$. With two properties above, a cryptographic hash function can be used to check if a message has been changed during transmission. In real-world application, a digital signature (which will be discussed in the next section) is usually signed on the hash of a message to ensure the data integrity and non-reputation at the same time.

4.1 Message-Digest Algorithm(MD)

4.2 Secure Hash Algorithm(SHA)

4.3 Hashed Message Authentication Code

4.4 Birthday Attack

Birthday Problem
Birthday Attack comes from the idea of Birthday Problem. The problem is to compute a probability that at least two people have the same birthday in a group of n people. Let A be the event that two people have the same birthday in a group of n people. The probability can be computed as:

$$P(A) = 1 - \prod_{i=1}^{n} \frac{365 - i + 1}{365}$$

With $n = 23$, the probability exceeds 50%. With $n = 70$, the probability is 99.9%.

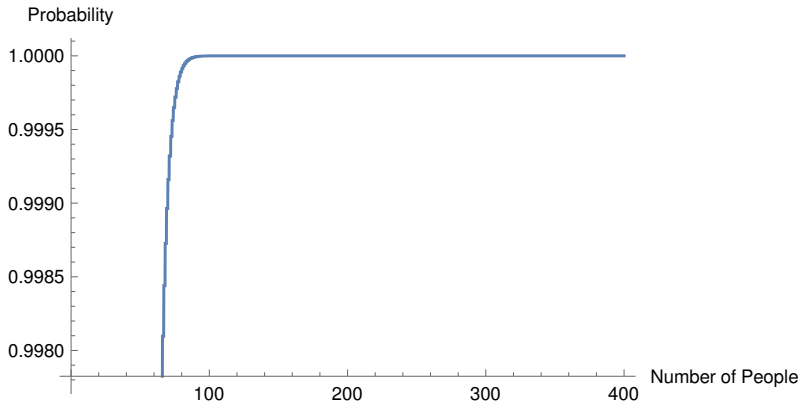

Figure 4.1: Probability of Birthday Problem

Chapter 5

Pseudorandom Number Generator

5.1 LFSR Pseudorandom Number Generator

5.1.1 Linear-Feedback Shift Register(LFSR)

Linear-Feedback Shift Register (LFSR) is defined as a linear recursive relation with certain length. I will illustrate a m-bit linear recursive relation of an LFSR below. The bits of the LFSR are generated with coefficients $c_0, c_1, \cdots, c_{m-1}$ and initial values $x_1, x_2, \cdots, x_m$.

$$x_{n+m} \equiv c_0 x_n + c_1 x_{n+1} + \cdots + c_{m-1} x_{n+m-1} \pmod 2$$

Here I illustrate a simple LFSR below.

Example 5.1.1. Let $X = (x_1, x_2, x_3) = (1, 1, 0)$ be initial values and $C = (c_0, c_1, c_2) = (1, 1, 1)$ be the coefficient. The LFSR generated with (X, C) are:

$$1100110011\cdots$$

5.2 B.B.S. Pseudorandom Number Generator

B.B.S stands for Blum Blum Shub which are the last names of the designers of the algorithm. First define a Blum prime.

Definition 5.2.1 (Blum Prime)**.** A prime p is a Blum prime if $p \equiv 3 \pmod 4$

Algorithm 5.1 B.B.S. Pseudorandom Number Generator

1: Choose two random Blum primes p, q. Compute $n = pq$.
2: Choose a random seed $s \in [1, n-1]$.
3: Compute $x_0 \equiv s^2 \pmod n$ and $z_0 \equiv x_0 \pmod 2$.
4: Iteratively compute $x_i \equiv x_{i-1}^2 \pmod n$ and $z_i \equiv x_i \pmod 2$.
5: Output the sequence $z_1, z_2, z_3, \cdots$.

Part II

Asymmetric Cryptography

Chapter 6

RSA Cryptosystem

RSA cryptosystem [1] is one of the first public key cryptosystem. The design of RSA algorithm is based on the difficulty of factorization of a composite number which is a product of two large primes. The public key pair in RSA algorithm is (n, e) where n is the product of two large primes and e is the encryption exponent.

6.1 RSA Algorithm

Suppose Bob wants to send Alice an encrypted message with RSA Cryptosytem. The process is illurstrated below:

KeyGen part of RSA algorithm is illustrated below.

Algorithm 6.1 RSA Algorithm - KeyGen

1: Alice chooses prime p, q, compute $n = pq$ and publish n.
2: Alice computes encryption exponent e : $gcd((p-1)(q-1), e) = 1$ and publish e.
3: Compute decryption exponent d : $de \equiv 1 \pmod{(p-1)(q-1)}$ and keep d secret.

Enc part of RSA algorithm is illustrated below.

Algorithm 6.2 RSA Algorithm - Enc

1: Bob encrypts plaintext m to ciphertext c : $c \equiv m^e \pmod{n}$ and sends c to Alice

Dec part of RSA algorithm is illustrated below.

Algorithm 6.3 RSA Algorithm - Dec

1: Alice decrypts ciphertext c to plaintext m : $m \equiv c^d \pmod{n}$.

6.1.1 Correctness of RSA algorithm

I will prove the correctness of RSA algorithm below. Consider that

$$c^d \equiv m^{ed} \pmod{n}$$

By the construction, we have that $de \equiv 1 \pmod{\phi(n)}$. Thus there exists $k \in \mathbb{Z}$ such that $de = 1 + k\phi(n)$. By substitution, we have that

$$c^d \equiv m^{ed} \equiv m^{1+kn} \equiv m \cdot m^{k\phi(n)} \equiv m \cdot (m^{\phi(n)})^k \pmod{n}$$

By the construction of n, we have that $\phi(n) = (p-1)(q-1)$. By Fermat's Little Theorem, we have that $m^{(p-1)(q-1)} \equiv 1 \pmod{p}$ and $m^{(p-1)(q-1)} \equiv 1 \pmod{q}$. By Chinese Remainder Theorem, we have that $m^{\phi(n)} \equiv m^{(p-1)(q-1)} \equiv 1 \pmod{n}$. By substitution, we have that

$$c^d \equiv m^{ed} \equiv m^{1+kn} \equiv m \cdot m^{k\phi(n)} \equiv m \cdot (m^{\phi(n)})^k \equiv m \cdot 1^k \equiv m \pmod{n}$$

Therefore, we proved the correctness of RSA algorithm.

6.1.2 Partition of messages to be encrypted

Let m be the integer representation of the message. If $m > n$, then we need to partition m into $m_1, m_2, \cdots m_k$ such that each $m_i < n$. The partition is required when $m > n$ since RSA algorithm performs arithmetic modulo n. If $m > n$, the decrypted message from the ciphertext may be different from the original plaintext.

Assume that $c^d \equiv m' \pmod{n}$. By modularity, if $m < n$, then we have that $m' = m$. However, if $m > n$, we will have $m' > m$ since $m' < n$. Thus the decrypted message will be different from the original message.

6.2 Attack on RSA

6.2.1 Short Plaintext

Suppose the modulus being used for encryption is n, and the plaintext message, represented as a number m,has k digits, where k is small,say less than 18. You intercept the ciphertext, a number denoted by c. You also have the public RSA encryption key (n, e),but not the decryption exponent d,and not the factorization of n.

To perform the attack, make two tables with all values of $cx^{-e} \pmod{n}$ where x varies from 1 to 10^9 say Table A, and the values of $y^e \pmod{n}$ where y varies from 1 to 10^9, say Table B. Compare each entry in two tables, if a match is found, then the attack is successful. Consider that we have :

$$cx^{-e} \equiv y^e \pmod{n}$$

Then

$$c \equiv xy^e \pmod{n}$$

By RSA encryption, we have that $c \equiv m^e \pmod{n}$. Thus it can be implied that

$$m \equiv xy \pmod{n}$$

In the worst case, the attack needs 2×10^9 times computations. It is reasonable that computers nowadays are able to do 2×10^9 computations in a reasonable time. It is notable that the attack can be used to derive the only plaintext. The secret keys can not be derived from this attack.

6.2.2 Continued Fraction Attack

TBD

6.2.3 Low-entropy Prime Number Attack

RSA keys incorrectly generated with random generator with low entropy can find the same prime [17]. This results in that one of the prime factors of different RSA keys is the same.

Consider $N = p \cdot q$ by RSA algorithm. We may assume that $N_1 = p \cdot q_1$ and $N = p \cdot q_2$ for two RSA keys. Since q_1, q_2 are prime numbers, it is obvious that $p = gcd(N_1, N_2)$. In this way, we can find q_1, q_2 for N_1, N_2.

6.3 Factorization Attacks

In this section, several factorization techniques will be discussed on the number theory basis. A factorization technique based on elliptic curves is also discussed in Appendix D.3.2.

6.3.1 P-1 Factorization

We will illustrate the $p-1$ factorization below. Suppose we need to factor $n = pq$ where p, q are distinct primes.

Algorithm 6.4 P-1 factorization

1: Choose random integer $a > 2$. Upper bound β.
2: Compute $b_1 \equiv a \pmod{n}$ and $b_j \equiv b_{j-1}^j \pmod{n}$ for $j = 2, 3, \cdots, \beta$.
3: Let $d = gcd(b_\beta, n)$. If $1 < d < n$, then d is a non-trivial factor of n. Else the factorization failed, choose another a or β.

Correctness of P-1 factorization

We will explain why $p-1$ factorization can factor $n = pq$ below. By Fermat's Little Theorem, $a^{p-1} \equiv 1 \pmod{p}$. Thus $gcd(a^{p-1} - 1, n)$ is a non-trivial factor of n. We need to find the integer m such that $p - 1 | m$ and $q - 1 \nmid m$ Suppose $p - 1 = \sum_{i=1}^{k} p_i^{e_i}$ and $q - 1 = \sum_{i=1}^{j} q_j^{e_j}$ with $p_{i-1} < p_i$ and $q_{i-1} < q_i$. Without loss of generality, assume $p_k < q_j$. Take the upper bound $p_k \leq \beta < q_j$, then $p - 1 | \beta!$ and $q - 1 \nmid \beta!$.

Construct of $n = pq$ to resist P-1 factorization

From explanation above, we know that the upper bound β must satisfy the condition that $p_k \leq \beta < q_j$. It provides us with two ideas on how we can construct $n = pq$ to resist $p-1$ factorization.

1. Let p, q have large prime factors say p_0, q_0. With large prime factors, the required β will also be large. Suppose that p_0, q_0 have 10^9 digits. Then it would be hard for a computer to factor $n = pq$ in limited time.

2. Let the difference between p_0, q_0 be relatively small. Suppose that $p - 1 > \beta!$, then it would be impossible to factor $n = pq$ with $p - 1$ factorization.

6.3.2 Fermat's Factorization

Fermat's factorization is a simple factorization method. The idea is based on the fact that an odd interger N can be represented as

$$N = a^2 - b^2 = (a + b)(a - b) \text{ where } a, b \in \mathbb{Z}$$

Fermat's factorization in its simplest form can be slow. However, with combination of trivial division, Fermat's factorization can be effective than both its simplest form and trivial division.

Algorithm 6.5 Fermat's Factorization - Simple Form

1: Let $a = \lceil \sqrt{n} \rceil$.
2: Compute $b = a^2 - N$.
3: Repeat Step 1,2 until b is a square.
4: N is factorized as $a - \sqrt{b}$ and $a + \sqrt{b}$

6.3.3 Pollard's Rho algorithm for factorization

Background for Pollard's Rho

We first discuss the computational background of Pollard's Rho Algorithm. Let S be a finite set and $f : S \to S$. Pick $x_0 \in S$ and let $x_i = f(x_{i-1})$. We make a sequence $x_0, x_1, x_2, \cdots$. Since S is finite, then we know that there exists $i < j$ such that $x_i = x_j$. Let $P = j - i$. Then $x_i = x_{i+p}$. Apply f, we have that

$$x_{i+1} = f(x_i) = f(x_{i+p}) = x_{i+1+p}$$

By induction, for all $k \geq 0$, $x_{i+k} = x_{i+k+p}$

Now we illustrate the Pollard's Rho Algorithm for Factorization. Assume that we need to factor $n = pq$ where p, q are distinct odd primes.

Algorithm 6.6 Pollard's Rho for Factorization

1: Pick initial value x_0 and function f.
2: Compute $f(x_{i-1}) \equiv x_i \pmod{n}$ for $i = 1, 2, \cdots$.
3: Let $d = gcd(x_i - x_j), n$. If $1 < d < n$, then d is a non-trivial factor of n.
4: If $d = n$, then the factorization failed. Try another x_0 or f.

Chapter 7

Cryptosystems with Discrete Logarithm

Please refer to Appendix B.1.9 for Discrete Logarithm Problem.

7.1 ElGamal Cryptosystem

ElGamal cryptosystem [2] is based on the difficulty of solving a discrete logarithm problem in a multiplicative group, say $\mathbb{Z}_p^\times$. The public key pair in ElGamal cryptosystem is (p, α, β) where p is a large prime and α is a primitive root modulo p.

Suppose Bob wants to send Alice an encrypted message with ElGamal Cryptosytem. The process is illurstrated below:

KeyGen part of ElGamal Cryptosystem is illustrated below.

Algorithm 7.1 ElGamal Cryptosystem - KeyGen

1: Alice Chooses a large prime p, and a primitive root α modulo p.
2: Alice Chooses a secret integer a such that $a \not\equiv 1 \pmod{p-1}$.
3: Alices Compute $\beta \equiv \alpha^a \pmod{p}$.
4: Alice publishes the public key tuple (p, α, β).

Enc part of ElGamal Cryptosystem is illustrated below.

Algorithm 7.2 ElGamal Cryptosystem - Enc

1: Bob chooses random integer k and compute $r \equiv \alpha^k \pmod{p}$.
2: With plaintext m, Bob computes $t \equiv \beta^k m \pmod{p}$.
3: Bob sends the encrypted text (r, t) to Alice.

Dec part of ElGamal Cryptosystem is illustrated below.

Algorithm 7.3 ElGamal Cryptosystem - Dec

1: With ciphertext (r, t), Alice computes $m \equiv tr^{-a} \pmod{p}$ to obtain plaintext m.

7.1.1 Correctness of ElGamal Cryptosystem

Now I will justify the correctness of ElGamal Cryptosystem. By the construction of ElGamal Cryptosystem, we have that the encrypted message is a 2-tuple (r, t) where $r \equiv \alpha^k \pmod{p}$ and $t = \beta^k m$. Consider the decryption operation

$$tr^{-a} \equiv \beta^k m \alpha^{-ak} \equiv \alpha^{ak} m \alpha^{-ak} \equiv m \pmod{p}$$

Thus we have proved the correctness of ElGamal Cryptosystem.

7.1.2 Randomness of integer k

Note that it is critical to choose a random k every time when using ElGamal Encryption. Suppose that m_1 and m_2 are plaintexts and they are encrypted with ElGamal Encryption with the same k. Assume that the attacker knows the plaintext m_1, then he could derive the plaintext of m_2. Let $t_1 \equiv \beta^k m_1 \pmod{p}$ and $t_2 \equiv \beta^k m_2 \pmod{p}$. Then we have that

$$t_1 m_1^{-1} \equiv \beta^k \equiv t_2 m_2^{-1} \pmod{p}$$

Therefore m_2 can be derived as:

$$m_2 \equiv t_2 (t_1 m_1^{-1})^{-1} \equiv t_2 m_1 t_1^{-1}$$

7.2 Diffie-Hellman key exchange

Diffie-Hellman Key Exchange is a key infrastructure protocol based on discrete logarithm problem. Diffie-Hellman Key Exchange is widely used in exchanging the secret key used in symmetric cryptosystems. Diffie-Hellman Key Exchange establishes the condition for secure communication without two parties meeting in person to exchange secret key.

Algorithm 7.4 Diffie-Hellman Key Exchange

1: Alice and Bob choose a large prime p and a primitive root a modulo p.
2: Alice chooses a secret integer i, send $a^i \pmod{p}$ to Bob.
3: Bob chooses a secret integer j, send $a^j \pmod{p}$ to Alice.
4: Alice compute $y \equiv (a^i)^j \pmod{p}$ to obtain the shared message.
5: Bob compute $y \equiv (a^j)^i \pmod{p}$ to obtain the shared message.

7.3 Attack Discrete Logarithm Problem

7.3.1 Baby Step Giant Step

We will illustrate the algorithm of Baby Step Giant Step below. Consider the discrete logarithm problem $g^k \equiv h \pmod{q}$, we give the *Baby Step Giant Step Algorithm* below.

Algorithm 7.5 Baby Step Giant Step

1: Take $m \in \mathbb{Z}^+$ and $m \geq \sqrt{N}$. Compute g^m. (Generally take $m = \lceil N \rceil$).
2: For $i = 0, 1, 2, \cdots, m-1$, compute g^i. (Baby Step Here)
3: For $j = 0, 1, 2, \cdots, m-1$, compute $h \cdot g^{-jm}$. (Giant Step Here)
4: $h \cdot g^{-jm} \equiv g^i \pmod{q}$ for some i, j. Since $h \equiv g^{jm+i} \pmod{q}$, $k = jm + i$.

7.3.2 Pollard's Rho algorithm for Discrete Logarithm

Consider the discrete logarithm problem $g^k \equiv h \pmod{q}$, we give the *Pollard's Rho Algorithm* below.

Algorithm 7.6 Pollard's Rho Algorithm for DL

1: Choose k bins where $3 \leq k \leq 20$ and k pairs of random integers $(r_i, s_i) \in \mathbb{Z} \times \mathbb{Z}$.
2: Let $m_i \equiv g^{r_i} + h^{s_i} \pmod{q}$.Let the function f be defined as $f(d) \equiv d \cdot m_i \pmod{q}$. where$d \equiv i \pmod{k}$.
3: Choose random integer $d_0 \equiv g^{r_0} \cdot h^{s_0} \pmod{q}$.
4: Iteratively compute $d_j \equiv f(d_{j-1}) \pmod{q}$ for $j = 1, 2, \cdots$. Write $d_j = g^{a_j} \cdot h^{b_j}$.
5: For so me i, j, $d_i \equiv d_j \pmod{q}$. Then $g^{a_j} \cdot h^{b_j} \equiv g^{a_i} + h^{b_i} \pmod{q}$.
6: Write $g^{(a_j - a_i)} \equiv h^{(b_i - b_j)} \pmod{q}$. Thus $a_j - a_i \equiv k(b_i - b_j) \pmod{|g|}$.
7: If $gcd(b_i - b_j, |g|) = 1$, solve for k.
8: Else let $e = gcd(b_i - b_j, |g|)$. Then $\frac{a_j - a_i}{e} \equiv k \cdot \frac{b_i - b_j}{e} \pmod{\frac{|g|}{e}}$, solve for k.

7.3.3 Pohlig-Hellman Algorithm

Consider the discrete logarithm problem $g^k \equiv h \pmod{q}$, we give the *Pohlig-Hellman Algorithm* below.

Algorithm 7.7 Pohlig-Hellman Algorithm

1: Compute $|g| \equiv p_1^{e_1} \cdot p_2^{e_2} \cdots p_n^{e_n} \pmod{q}$.
2: For each p_i, compute $\tilde{g} \equiv g^{\frac{|g|}{p}} \pmod{q}$. Iteratively compute $\tilde{g}^2 \cdots \tilde{g}^{p_i} \pmod{q}$.
3: Compute $h^{\frac{|g|}{p_i}} \equiv \tilde{g}^j \pmod{q}$.
4: Write $k_i \equiv j \pmod{p_i}$.
5: Use Chinese Remainder Theorem to compute k with k_i.

Chapter 8

Digital Signature

A digital signature is used to check if a message comes from a certain user i.e., non-reputation of a user. Say a signer with key pair (P_k, S_k). He signs a message m to create a signature sig with his secret key by computing $Sign_{S_k} = sig_{S_k}$. To verify the signature, a verifier computes $Verify_{P_k} = m$ with the public key of the signer to check if the message comes from the signer. Usually, a cryptographic hash function is applied with digital signature. The signer sign on the hash of the message to show that the message is not changed during transmission and the message comes from him by computing $Sig_{S_k}(H(m)) = sig_{S_k}$ where H is the cryptographic hash function. Attacks on digital signatures are usually to create a collision of cryptogrpahic hash function such that the attacker can trick the signer to sign on a malicious message which has the same hash value with the original message.

8.1 RSA Signature

RSA is also used in digital signature which is used to verify the identity of sender and guarantee non-reputation of sender. Instead of using public key to encrypt, RSA signature uses secret key to sign a certain message. However, the secret key of sender can not be derived from the signature as it is a discrete logarithm problem to derive the sender's secret key from the plaintext and signatures which is of great difficulty.

Algorithm 8.1 RSA Signature

1: Alice chooses prime p, q, compute $n = pq$ and publish n.
2: Alice computes public key e : $gcd((p-1)(q-1), e) = 1$.
3: Alice computes secret key d : $de \equiv 1 \pmod{(p-1)(q-1)}$.
4: Alice signs plaintext m to signature s : $s \equiv m^d \pmod{n}$ and sends s to Bob.
5: Bob verifies the signature s : $m \equiv s^e \pmod{n}$.

The correctness of RSA Signature is proved the same as the proof of correctness of RSA algorithm. Please refer to Section 6.1.1 for details.

8.2 ElGamal Digital Signautre

Algorithm 8.2 ElGamal Digital Signature

1: Choose a large prime p, and a primitive root α modulo p.
2: Choose a secret integer a such that $a \not\equiv 1 \pmod{p-1}$.
3: Compute $\beta \equiv \alpha^a \pmod{p}$.
4: The public key is the tuple (p, α, β).
5: Choose random integer k such that $gcd(k, p-1) = 1$.
6: Compute $r \equiv \alpha^k \pmod{p}$.
7: Compute $s \equiv k^{-1}(m - ar) \pmod{p-1}$ where m is the message.
8: The signature is the triple (m, r, s).
9: To verify the signature, compute $v_1 \equiv \beta^r r^s \pmod{p}$ and $v_2 \equiv a^m \pmod{p}$
10: If $v_1 \equiv v_2 \pmod{p}$, the signature is valid.

8.3 Blind Signature

Blind signature is a type of signature in which the content of the message is blinded before the signature. A blind signature is usually employed in a privacy-related scenarios in which signer and the message author are different parties such as in cryptocurrency. By employing a blind signature, the message author can obtain a signature from the signer without disclosing the content of the message. However, a blind signature can be risky for the signing authority since it can issue a valid signature without knowing the content of the message.

8.3.1 RSA Blind Signature

Assume that Arthur is a signing authority with a RSA key pair $< (n, e), d >$. Suppose Bob wants Arthur to sign her message but he does not want Arthur to learn the content of the message. Bob can perform the following steps to gain the signature from Arthur.

Algorithm 8.3 RSA blind signature

1: Bob chooses a random $k \pmod{n}$ such that $gcd(k, n) = 1$. Bob computes $t \equiv k^e m \pmod{n}$ and sends t to Arthur.
2: Arthur computes $s \equiv t^d \pmod{n}$ to sign the message and sends to Alice.
3: Bob computes $m^d \equiv sk^{-1} \pmod{n}$ to obtain the signature.

The correctness of a RSA blind signature can be proved easily. Consider that

$$sk^{-1} \equiv t^d k^{-1} \equiv k^{ed} m^d k^{-1} \equiv k m^d k^{-1} \equiv m^d \pmod{n}$$

Chapter 9

Elliptic Curves Cryptography (ECC)

Elliptic Curves Cryptography is mainly based on discrete logarithm problem on elliptic curvers over finite fields. Say for $P, Q \in E(\mathbb{F}_p)$ such that $[m] \cdot P = Q$ for some $m \in \mathbb{Z}$. The problem is to find m. The basic knowledge of elliptic curves is discussed in Appendix D.

9.1 Encoding of message to a point on elliptic curve

I will illustrate the algorithm to encoding a message m to a point P on elliptic curve. It is assume that the elliptic curve is of Short Weierstrass Equation $E : y^2 = x^3 + Ax + B$ where $A, B \in \mathbb{F}_p$ where p is prime. Also, assume that $0 \leq m \leq \frac{p}{100} - 1$. Note that if m is greater than the upper bound, partition m into pieces to encrypt and concentrate the ciphertext then.

Algorithm 9.1 Encoding of Message in Elliptic Curve Cryptography

1: To encode : try $x_i = 100m + i$ for $0 \leq i < 100$ until $y_i^2 = x_i^3 + Ax_i + B \pmod p$. Then $P = (x_i, y_i)$
2: To decode : compute $m = \lfloor \frac{x_i - i}{100} \rfloor$

9.2 Massey-Omura Cryptosytem

Massey-Omura Cryptosystem[4] is a three-pass protocol. Three handshakes between two parties happened during the transmission of a message.

The public information is : $E, \mathbb{F}_q, N$.

The secret information is : a, b.

Suppse that Alice wants to send Bob an encrypted message using Massey-Omura Cryptosystem. The process is illurstated below:

KeyGen part of Massey-Omura Cryptosytem is illustrated below.

Algorithm 9.2 Massey-Omura Cryptosystem - KeyGen

1: Alice and Bob agree on an elliptic curve E over finite field $\mathbb{F}_q$. Let $N = |E(\mathbb{F}_q)|$, $P = (x, y)$ which represents the message m.
2: Alice and Bob each secretly pick random a and b such that $gcd(a, N) = 1$ and $gcd(b, N) = 1$.
3: Alice and Bob compute $a^{-1}, b^{-1} \in \mathbb{Z}^+$.

Enc part of Massey-Omura Cryptosytem is illustrated below.

Algorithm 9.3 Massey-Omura Cryptosystem - Enc

1: Alice computes $[a] \cdot P$, and sends to Bob.
2: Bob computes $[b] \cdot ([a] \cdot P)$, and sends to Alice.

Dec part of Massey-Omura Cryptosytem is illustrated below.

Algorithm 9.4 Massey-Omura Cryptosystem - Dec

1: Alice computes $[a^{-1}] \cdot ([b] \cdot ([a] \cdot P))$, and sends to Bob.
2: Bob compute $[b^{-1}] \cdot ([a^{-1}] \cdot ([b] \cdot ([a] \cdot P))) = m$

9.3 ElGamal Cryptosystem on elliptic curves

The ElGamaml Cryptosystem[2] by scalar multiplication of points on elliptic curves is illustrated below.

The public information is : $E, \mathbb{F}_q, P, Q$
The secrete information is : s, a

Algorithm 9.5 ElGamal Cryptosystem with Elliptic Curves

1: Bob picks an elliptic curve E over $\mathbb{F}_q$, a point $P \in E(\mathbb{F}_q)$ and random s. Compute $Q = [s] \cdot P$.
2: For Alice, to send a message $m \in E(\mathbb{F}_q)$. She picks a random number a, and compute $[a] \cdot P$ and $m + [a] \cdot P$, then sends them to Bob.
3: Bob computes $(m + [a] \cdot Q) - [s] \cdot ([a] \cdot P)$ to decrypt.

The correctness of ElGamal Cryptosystem with elliptic curves is justified as below :

$$\begin{aligned}(m + [a] \cdot Q) - [s] \cdot ([a] \cdot P) &= m + [a] \cdot ([s] \cdot P) - [s] \cdot ([a] \cdot P) \\ &= m\end{aligned}$$

9.4 Diffie-Hellman Key Exchange on elliptic curves

The Diffie-Hellman Key Exchange[3] by scalar multiplication of points on elliptic curves is illustrated below.

The public information is : $P, E(\mathbb{F}_q)$
The secret information is : a, b

Algorithm 9.6 Diffie-Hellman Key Exchange in Elliptic Curve Cryptography

1: Alice and Bob pick a prime power q, an elliptic curve E over $\mathbb{F}_q$ and a point $P \in E(\mathbb{F}_q)$.

2: Alice chooses $a \in \mathbb{Z}$, Bob chooses $b \in \mathbb{Z}$.
3: Alice computes $[a] \cdot P$ and sends to Bob.
4: Bob computes $[b] \cdot P$ and sends to Alice.
5: Alice computes $[a] \cdot ([b] \cdot P))$.
6: Bob computes $[b] \cdot ([a] \cdot P))$.

9.5 Attack Discrete Logarithm Problems of Elliptic Curves

Let E be elliptic curve over $\mathbb{F}_q$, say $E(\mathbb{F}_q)$. Let $P, Q \in E(\mathbb{F}_q)$. The discrete logarithm problem of elliptic curve is to find $k \in \mathbb{Z}^+$ such that $kP = Q$. Several algorithms to solve discrete logarithm problem of elliptic curves will be illustrated below.

9.5.1 Baby Step Giant Step

The algorithm of Baby Step Giant Step is illustrated below.

Algorithm 9.7 Baby Step Giant Step For Elliptic Curves

1: Take $m \in \mathbb{Z}^+$ and $m \geq \sqrt{N}$. Compute mP. (Generally take $m = \lceil N \rceil$).
2: For $i = 0, 1, 2, \cdots, m-1$, compute iP. (Baby Step Here)
3: For $j = 0, 1, 2, \cdots, m-1$, compute $Q - jmP$. (Giant Step Here)
4: $Q - jmP = iP$ for some i, j. Since $Q = (jm + i)P$, $k = jm + i$.

Baby Step Giant Step takes $\sqrt{N}$ storage to solve the discrete logarithm problem. Thus generally take $N = q + 1 + 2\sqrt{q}$,which is the Hasse's Bound.

Existence of (i, j) in Baby Step Giant Step

I will show the correct of Baby Step Giant Step by showing the pair (i, j) must exist. Consider that we have $0 \leq k < N$. By Division Algorithm, we must have that $\exists q, r \in \mathbb{Z}$ such that $k = mq + r$ where $0 \leq r < m$. Thus

$$q = \frac{k - r}{m} \leq \frac{k}{m} \leq \frac{N}{m} \leq \sqrt{N} < m$$

Take $q = j, r = i$, we showed that (i, j) must exist and $0 \leq i, j < m$.

9.5.2 Pollard's Rho Algorithm

Computational background of Pollard's Rho Algorithm was discussed in the section *Pollard's Rho Algorithm* of Number Theory. Here the Pollard's Rho Algorithm for discrete logarithm problem of elliptic curves is illustrated.

Algorithm 9.8 Pollard's Rho Algorithm for DL in Elliptic Curves

1: Choose k bins where $3 \leq k \leq 20$ and k pairs of random integers $(r_i, s_i) \in \mathbb{Z} \times \mathbb{Z}$.
2: Let $M_i = r_i P + s_i Q$.Let the function $f : E(\mathbb{F}_q) \to E(\mathbb{F}_q)$ be defined as $f(R) = R + M_i$ where $R = (x, y) \in E(\mathbb{F}_q)$ and $x \equiv i \pmod{k}$.
3: Choose random point $R_0 = r_0 P + s_0 Q$ where $R_0 \in E(\mathbb{F}_q)$.
4: Iteratively compute $R_j = f(R_{j-1})$ for $j = 1, 2, \cdots$. Write $R_j = a_j P + b_j Q$.
5: For some i, j, $R_i = R_j$. Then $a_j P + b_j Q = a_i P + b_i Q$.
6: Write $(a_j - a_i)P = (b_i - b_j)Q$. Thus $a_j - a_i = k(b_i - b_j) \pmod{|P|}$.
7: If $gcd(b_i - b_j, |P|) = 1$, solve for k.
8: Else let $d = gcd(b_i - b_j, |P|)$. Then $\frac{a_j - a_i}{d} \equiv k \cdot \frac{b_i - b_j}{d} \pmod{\frac{|P|}{d}}$, solve for k.

9.5.3 MOV attack

The idea of MOV attack is to reduce discrete logarithm problem of elliptic curves to discrete logarithm problem in $\mathbb{F}_{q^s}^{\times}$ since there may be easier way to solve in $\mathbb{F}_{q^s}^{\times}$ such as Index Calculus. Let $n = |P|$, we want to use Weil Pairing e_n and need to move up for $\mathbb{F}_{q^s}$ such that $E(\mathbb{F}_{q^s})[n] \cong \mathbb{Z}_n \times \mathbb{Z}_n$ (Assume that $char(\mathbb{F}_q) \nmid n$). The MOV attack is illustrated below.

Algorithm 9.9 MOV attack

1: Pick random $T \in E(\mathbb{F}_{q^s})$.
2: Let $m = |T|$, $d = gcd(m, n)$.
3: Let $T' = (m/d)T$, now $|T'| = d$.
4: Note $e_n(Q, T') = e_n(kP, T') = e_n(P, T')^k$. Compute $e_n(P, T'), e_n(Q, T')$.
5: Solve the discrete logarithm problem of $e_n(P, T')^k$ in $\mathbb{F}_{q^s}^{\times}$.
6: Get k modulo order of $e_n(P, T')$, which is a divisor of d.
7: If needed, repeat with new random T.

Part III

Applied Cryptography

Chapter 10

Cryptographic Protocols

10.1 Kerberos

Chapter 11

Attack to Cryptographic Protocols

11.1 Man-In-Middle Attack

A Man-In-Middle Attack is an attack in which the attacker tries to listen to or alter the content of communication between two parties by pretending the attacker's key as the receiver's key and the sender's key. Two parties involved in the communication believe that they are communicating with each other but they are communicating with the attacker. An example is illustrated below.

Example 11.1.1 (Man-In-Middle Attack). Assume that Alice and Bob would like to communicate with each other while Eve is the attacker who wants to intercept their communication. Alice has the public key P_A, Bob has the public key P_B, and Eve has the public key P_E.

1. Alice sends P_A to Bob but intercepted by Eve.
2. Eve sends P_E to Alice pretending that she is Bob.
3. Eve sends P_E to Bob pretending that she is Alice.
4. Alice thinks P_E is the key from Bob and Bob thinks P_E is the key from Alice.

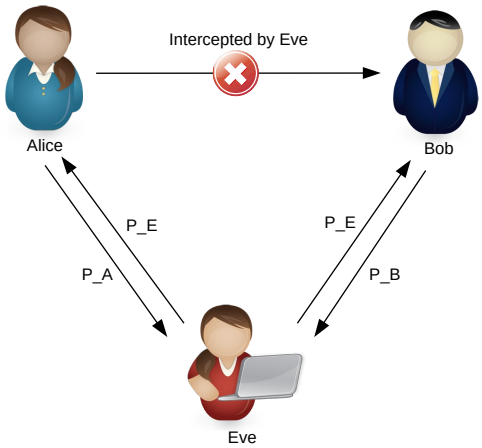

Figure 11.1: Man-In-Middle Attack

Certificate Authority

A certificate authority (CA) provides a way to prevent the Man-In-Middle Attack. Assume that a certificate authority is a trusted third party. When a user requests a public key from a CA, the CA provides the CA's signature on the public key to show that the public key belongs to a certain party.

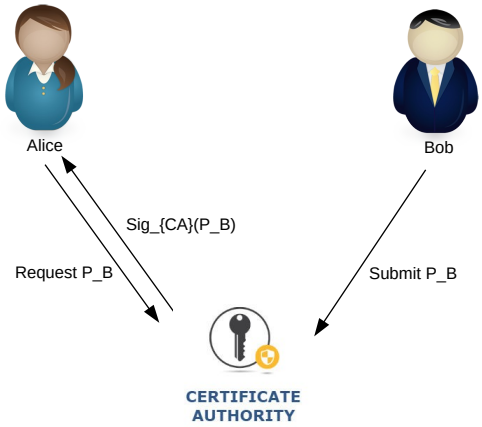

Figure 11.2: Application of Certificate Authority

Part IV

Advanced Topics in Cryptography

Chapter 12

Homomorphic Encryption

Homomorphic Encryption allows computation on ciphertexts. After decryption, the final message obtained matches the result of operations performed on plaintext. A homomorphic encryption scheme is defined below.

Definition 12.0.1 (Homomorphic Encryption)**.** A Homomorphic Encryption is a public key encryption ϵ scheme with three functions:

$$KeyGen_\epsilon, Encrypt_\epsilon, Decrypt_\epsilon$$

Let Pk be the public key, C be a circuit which is operation on the text and $\Psi =< \psi_1, \psi_2, \cdots, \psi_n >$ be ciphertexts. Let $< \pi_1, \pi_2, \cdots, \pi_n >$ be plaintexts such that $\psi_i = Encrypt_\epsilon(P_k, \pi_i)$ for all i. With the circuit C, the homomorphic encryption scheme could achieved the following property:

$$\zeta = C(\psi_1, \psi_2, \cdots, \psi_n) \Rightarrow C(\pi_1, \pi_2, \cdots, \pi_n) = Decrypt_\epsilon(Sk, \zeta)$$

Several cryptosystems can achieve Partially Homomorphic Encryption with group operations. For example, RSA and ElGamal can utilize Homomorphic Encryption under modular multiplication since they are designed on the multiplicative groups.

12.1 Fully Homomorphic Encryption

The first Fully Homomorphic Encryption(FHE) Scheme is designed by Gentry using ideal lattice [12]. Gentry's scheme supports addition and multiplication. The idea of Gentry's scheme will be illustrated below.

Chapter 13

Ring Signature

The Ring Signature was invented by Ron Rivest, Adi Shamir, and Yael Tauman in [15]. Any user who has a key in a group can perform the signature. The signature can be verified with public key from each user in the group. Ring Signatures provide certain anonymity for the user who signs the message since it is computationally infeasible to determine the user who signs the message in the group. Define a Ring Signature as below.

Definition 13.0.1 (Ring Signature)**.** Assume that there are n people in a group. Each member i in the group has the key pair (P_i, S_i). A ring signature scheme includes two function:

$$Sig_{S_i}, Verify_{P_j}$$

where $i, j = 1, 2, \cdots, n$. A user with key pair (P_i, S_i) computes $Sig_{S_i}(m) = sig$ to create the signature sig from message m. A verifier computes $Verify_{P_j}(sig) = m$ where $j = 1, 2, \cdots, n$.

The ring signatures based on RSA will be illustrated below.

13.1 RSA Ring Signature

Chapter 14

Secure Multi-party Computation (MPC)

Secure Multi-party Computation (MPC) is to create a scheme for multiple parties to compute a function with their inputs while keeping their input secret. Assume that $< x_1, x_2, \cdots, x_n >$ are inputs of different users and $f(x_1, x_2, \cdots, x_n)$ is the function to be computed. Secure Multi-party Computation is to find f with secret $< x_1, x_2, \cdots, x_n >$.

Consider a simple example. Suppose Alice, Bob and Charlie respectively have a salary of amount x, y, z. They would like to find out the highest of the three salaries without revealing the amount of their own salary. Following the model we illusrtated above, x, y, z are the secret inputs and $f(x, y, z) = max(x, y.z)$ is the result they would like to compute.

Secure Multi-party Computation is firstly introduced as Secure Two-party Computation (2PC) by Andrew Yao [13]. The protocol to solve the 2PC problem is through the garbled circuit. At a high level, the garbled circuit constructs an oblivious transfer which keeps the data secret. Compared with 2PC, protocols for MPC mainly make use of secret sharing include Shamir's Secret Sharing (SSS) [14] and additive secret sharing. 2PC and MPC are critical in designing consensus protocol in the distributed network for the reason that Zero-knowledge Proof can be modeled as a 2PC problem and distributed voting schemes rely heavily on MPC.

14.1 Yao's Garbled Circuit

Chapter 15

Zero-knowledge Proof

In a zero-knowledge proof(ZKP), there are basicailly two parties Prover and Verifier. Prover intends to prove that he has some knowledge but the prover does not want to reveal the content of the knowledge while the verifier wants to verify that the prover does know the knowledge. Compared with MPC we discussed in the previous section, ZKP focuses on the proving and verifying process instead of obtaining a shared result for the both parties involved.

15.1 The Abstract Example

15.2 Feige-Fiat-Shamir identification scheme

Feige-Fiat-Shamir identification scheme is a parallel zero-know proof process. It is developed by Uriel Feige, Amos Fiat, and Adi Shamir. Assume Peggy (the prover) has a sequence of keys, say $s_1, s_2, \cdots, s_k.$, and she wants to prove to Victor(the verifier) that she has the keys but she does not want Victor to learn the content of the keys.

Algorithm 15.1 Feige-Fiat-Shamir identification scheme

1: For each s_i, Peggy computes $v_i \equiv s_i^{-2} \pmod{n}$ and sends v_i to Victor.
2: Peggy chooses a random iteger r, computes $x \equiv r^2 \pmod{n}$ and send x to Victor.
3: Victor chooses random $b_1, b_2 \cdots, b_k$ where $b_i \in \{0, 1\}$ and sends b_i to Peggy.
4: Peggy computes $y \equiv r s_1^{b_1} s_2^{b_2} \cdots s_k^{b_k} \pmod{n}$ and sends y to Victor.
5: Victor checks if $x \equiv y^2 v_1^{b_1} v_2^{b_2} \cdots v_k^{b_k} \pmod{n}$.
6: Repeat 1-4 several times.

15.3 zk-SNARK

zk-SNARK is a type of Zero-knowledge proof construction. The full name of zk-SNARK is "Zero-knowledge Succinct Non-interactive Argument of Knowledge". "Succinct" and "Non-interactive" are two important properties in this type of proof construction. "Succinct" means the proofs can be verified within a few milliseconds with a short proof length. "Non-interactive" means prover and verifier does not have to communicate between two parties for multiple rounds. We will give a high-level description of zk-SNARK below.

15.3.1 High-Level Description

A zk-SNARK can be used to prove and verify such a statement: *"given public known input v and public predict F, proofer knows a secret input w such that $F(v, w) = true$.* A zkSNARK involves three parties: setup, prover, verifier. Each party executes their algorithm respectively.

- setup is a trusted third party. It receives the predict F and outputs proving key pk_F and verification key vk_f. Note both pk_F and vk_F are public. Although keys are made public, some secret values are involved during the intermediate computations. Thus setup must be a trusted third party.
- prover receives the proving key pk_F, the public input v and the secret input w. With inputs and key listed above, prover computes and outputs the proof π.
- verifier receives the verification key vk_F, the public input v and the proof π. verifier outputs the decision to accept or reject the proof.

15.3.2 R1CS Instance

R1CS stands for rank-one constraint satisfcation. An **R1CS** ϕ is a 6-tuple $(\mathbb{F}, k, N, M, \mathbf{a}, \mathbf{b}, \mathbf{c})$. Specifically, $\mathbb{F}$ is a finite field. k is the number of input. N is the number of variables $(k \leq N)$. M is the number of constraints. **a,b,c** are $(1 + N) \times M$ matrices over the finite field $\mathbb{F}$.

An input-witness pair (v, w) where $v \in \mathbb{F}^k$ and $w \in \mathbb{F}^{N-k}$ satisfies that

$$\forall j \in [M], (\sum_{j=0}^{N} \mathbf{a}_{i,j} z_i) \cdot (\sum_{j=0}^{N} \mathbf{b}_{i,j} z_i) = (\sum_{j=0}^{N} \mathbf{c}_{i,j} z_i)$$

where $z \in \mathbb{F}^{N+1}$ is the concatenation of 1, v, and w.

15.3.3 zk-SNARK with R1CS

We combine R1CS instance and the high-level description we discussed previously to give a formal definition of a zk-SNARK interface. A zkSNARK involves three parties: setup, prover, verifier. Each party executes their algorithm $\mathcal{S}, \mathcal{P}, \mathcal{V}$ respectively.

- setup. On the input of a R1CS instance $\phi = (\mathbb{F}, k, N, M, \mathbf{a}, \mathbf{b}, \mathbf{c})$, $\mathcal{S}$ outputs the proving key pk and the verification key vk.
- prover. On the input of the proving key pk, input $v \in \mathbb{F}^k$, and witness $w \in F^{N-k}$. $\mathcal{P}$ outputs the proof π.
- verifier. On the input of the verification key vk, input $v \in \mathbb{F}^k$, and the proof π. $\mathcal{V}$ outputs the decision to accept or reject the proof.

15.3.4 zk-SNARK properties

Let ϕ be an R1CS instance, let pk, vk be the key pair generated by setup on the input ϕ. zk-SNARK holds following properties.

- *Completeness.* For every input-witness pair (v, w) which satisfies ϕ, the proof $\phi = \mathcal{P}(pk, v, w)$ results in $\mathcal{V}(vk, v, \pi) = 1$.

- *Soundness.* For every input-witness pair (v, w) which does not satisfy ϕ, no malicious prover can produce the valid proof $\phi = \mathcal{P}(pk, v, w)$ which results in $\mathcal{V}(vk, v, \pi) = 1$.
- *Zero knowledge.* For every input-witness pair (v, w) which satisfies ϕ, the proof $\phi = \mathcal{P}(pk, v, w)$ reveals no information about the witness w.
- *Succcinctness.* The proof π has size $\mathcal{O}(1)$ and $\mathcal{V}$ has running time $\mathcal{O}(k)$.

Chapter 16

Secret Sharing

Suppose a missile needs to be activiated with a secret token. The token is divided into 7 parts and each part is assigned to a senior officer such that one senior officer is not able to perform a launch sololy. However, in the case of a war, the token must be able to be reconstructed by 3 senior officers. In this chapter, secret sharing schemes to utilize this property will be discussed.

16.1 Secret Spliting

Secret Spliting is the simplest situation. Consider that you want Alice and Bob to share a message M(assume M is represented as an integer) such that they must cooperate to reconstruct the message. A solution can be:

1. Choose an integer n such that for all possible M, n > m
2. Choose a random integer $r \pmod{n}$.
3. Assign r to Alice.
4. Assign $M - r$ to Bob.

It is necessary to have the random integer we choose modulo n since each integer mod n is assigned with equal probability $\frac{1}{n}$.

In more general cases, consider we need to share a message among m people. We simply choose $m - 1$random integers $r_1, r_2, \cdots r_{m-1} \pmod{n}$ and assign them to $m - 1$ of the people. Then we assign $M - \sum_{i=1}^{m-1} r_i \pmod{n}$ to the remaining person.

16.2 Threshold

Chapter 17

Blockchain and Cryptocurrency

Blockchain was first conceptualized in [16] by Satoshi Nakamoto in 2008. Blockchain is a cryptographic-secure decentralized ledger which is usually used to record transactions happened between nodes. With cryptographic hash function and timestamps, blockchain achieves the property that a modification on certain data will result in the change of the whole blockchain. Simultaneously, a blockchain is implemented in a decentralized (or distributed) system. Thus only the blockchain held by majority of the nodes is accepted. Currently, blockchain is implemented in different aspects including cryptocurrency, smart contract, and DeFi. In this chapter, severl blockchain models and their properties will be discussed. Some applications of blockchains will also be discussed in this chapter.

17.1 Bitcoin and Proof-of-Work

Bitcoin is the first cryptocurrency which utilizes blockchain structure. Bitcoin was first introduced by Satoshi Nakamoto in the paper "Bitcoin: A Peer-to-Peer Electronic Cash System" [16]. Bitcoin has important properties including pseudo-anonymity and decentralization. In this section, we will discuss the application of blockchain as cryptocurrency with the example of Bitcoin from different components of Bitcoin network including Bitcoin's transaction mode, block structure, and consensus mechanism.

17.1.1 Transaction mode

Address

In Bitcoin, each user holds a public-private ECDSA key pair, say $(Pub_A, Priv_A)$. The public key of a user is hashed by SHA-256 and RIPEMD-160 to become the address of the user in the Bitcoin network i.e.,

$$Addr_A = \mathsf{Version\_Byte}||\mathsf{RIPEMD160}(\mathsf{SHA256}(Pub_A))$$

The address is public in the Bitcoin network. Addresses indicates the input and output of a transaction. The pseudo-anonymity is achieved by using addresses since no real-world idenity is disclosed in the network. Bitcoin clients can generate a new address every time for a transaction to resist clustering heuristics and achieve a better anonymity.

Initiate a transaction

17.1.2 Block Structure

17.1.3 Proof-of-Work blockchain

17.2 Proof-of-Stake

Chapter 18

Quantum Techniques in Cryptography

Quantum techinques are being explored to establish secure communcation channel and to increase computating effciency.Thus two concepts which are *Quantum Cryptography* and *Post-quantum Cryptography* are developed. Quantum cryptography refers to the idea of exploiting quantum mechanics to perform some cryptographic tasks. Examples includes quantum key distrubition, quantum commitment, and quantum coin flipping. Post-quantum cryptography refers to the science of constructing cryptographic infrastructures which resist the attacks of quantum computation. In this chapter, we will discuss both Quantum Cryptography and Post-quantum Cryptography. For Quantum Cryptography, quantum key distribution will be discussed. For post-quantum cryptography, we will discuss Shor's Algorithm ,and its application for factorization of large composite primes and solving discrete logarithm problems.

18.1 Modeling of photon polarization

18.2 BB84 protocol

BB84 Protocol is a quantum key distribution (QKD) protocol. It uses photon polarization states to produce a random secret key which will be shared by two parties of the communication. There are two communication channels in BB84 protocol: Quantum Channel and classical Channel. Quantum channel is used to transmit polarized photons (the channel should not the state of polarized photons during transmission) and the classical channel is used to transmit ordinary messages. Suppose Alice and Bob would like to establish a key exchange using BB84 protocol. They can do the following steps.

1. Alice encodes the secert key sequence with randomly chosen basis for each bit. There are two bases $B_1 = \{|\uparrow\rangle, |\rightarrow\rangle\}$ and $B_2 = \{|\nwarrow\rangle, |\nearrow\rangle\}$. She encodes $0, 1$ to elements in B_1, B_2.
2. Alice sends the encoded sequence through the quantum channel.
3. Bob randomly chooses either B_1 or B_2 to measure the photons from Alice.
4. Bob tells Alice the basis he chooses for each bit through the classical channel.
5. Alice responds to Bob by telling which bases are correct.

6. Bod decodes the photons with correct bases.

Consider that two bases are used, Alice and Bob are expected to agree on half of the amount of bits Alice sends.

We will use a simple example to illurstate the process.

Example 18.2.1. Suppose Alice wants to send Bob the sequence $0, 1, 1, 0, 1, 1, 0, 1, 1$. The bases Alice chooses is $B_1, B_2, B_2, B_1, B_1, B_2, B_1, B_2, B_1$.

1. Alice encoded the sequence to: $|\uparrow\rangle, |\nearrow\rangle, |\nearrow\rangle, |\uparrow\rangle, |\rightarrow\rangle, |\nearrow\rangle, |\uparrow\rangle, |\nearrow\rangle, |\rightarrow\rangle$.
2. Bob tells Alice the sequence $B_1, B_1, B_2, B_2, B_2, B_2, B_1, B_2, B_2$ he chooses.
3. Alice responds to Bob with indeices of the correct bases i.e., $1, 3, 6, 7, 8$.
4. Bob decodes the photon sequence $|\uparrow\rangle, |\nearrow\rangle, |\nearrow\rangle, |\uparrow\rangle, |\nearrow\rangle$ to $0, 1, 1, 0, 1$.
5. The shared key is $0, 1, 1, 0, 1$.

18.3 Shor's Algorithm

Part V

Appendices

Appendix A

Notations

The following notations will be used in definitions and theorem in this note :

Important Sets

$\mathbb{Z}$	set of all integers.
$\mathbb{Q}$	set of all rational numbers.
$\mathbb{R}$	set of all real numbers.
$\mathbb{C}$	set of all complex numbers.
$\mathbb{N}$	$\{a \in \mathbb{Z} : a \geq 0\}$.
$\mathbb{Z}^+$	$\{a \in \mathbb{Z} : a > 0\}$.
$\mathbb{Q}^+$	$\{a \in \mathbb{Q} : a > 0\}$.
$\mathbb{R}^+$	$\{a \in \mathbb{R} : a > 0\}$.
$\mathbb{Q}^\times$	Multiplicative Group of $\mathbb{Q}$).
$\mathbb{R}^\times$	Multiplicative Group of $\mathbb{R}$.
$\mathbb{C}^\times$	Multiplicative Group of $\mathbb{C}$.
$\mathbb{Z}_n$	Congruence class modulo n.
$M_n(R)$	$\{n \times n$ matrices : entries from R$\}$ where $R = \mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{C}, etc.$
$GL_m(R)$	$\{A \in M_n(R) : det(A) \in \mathbb{R}^\times\}$.
$\mathbb{F}_q$	finite field with q elements.
$E(K)$	Elliptic Curve E over field K.
$\overline{K}$	Algebraic closure of field K.
R/I	ring R modulo ideal I.
$Aut(K)$	Automomorphism in field K.
$Gal(R/I)$	Galois Automomorphism in quotient ring R/I.

Relations

$a \mid b$	a divides b.
$a \equiv b \pmod{n}$	a is congruent to b modulo n
$G_1 \cong G_2$	G_1 is isomorphic to G_2.
$[x_1 : y_1 : z_1] \sim [x_2 : y_2 : z_2]$	$[x_1 : y_1 : z_1]$ is equal to $[x_2 : y_2 : z_2]$ in projective coordinate.

Special Elements in Group

1_R	Multiplicative identity in ring R,
0_R	Additive identity in ring R.
e_G	Identity in group G.
$\mathscr{O}$	Point at infinity of elliptic curves.

Appendix B

Number Theory

This section introduces the Number Theory knowledge which is required for cryptography.

B.1 Basic Concept of Number Theory

This section introduces basic concept of the Number Theory.

B.1.1 Division

We first introduce the Division Algorithm.

Theorem B.1.1 (Division Algorithm)**.** $\forall a, b \in \mathbb{Z}$ *and* $a \geq b, \exists! q, r \in \mathbb{Z} s.t. a = bq + r$ *where* $0 \leq r < |b|$.

Usually, the Division Algorithm is introduced for $a, b \in \mathbb{Z}^+$. However, it should be noted that Division Algorithm holds for arbitrary $a, b \in \mathbb{Z}$ i.e.,a, b may be negative.

With the Division Algorithm, we introduce the definition of a divides b below :

Definition B.1.1 (Division and Divisor)**.** Let $a, b \in \mathbb{Z}$, a divides b if $\exists q \in \mathbb{Z}$ such that $a = qb$, written as $a|b$. Then we say that b is a divisor(factor) of a.

B.1.2 Prime Numbers

Prime Number is a important concept in Number Theory and Cryptography. We give the definition of prime number below.

Definition B.1.2 (prime number)**.** Let $p \in \mathbb{Z}$. p is prime if the only divisors of p are 1 and p.

B.1.3 Greatest Common Divisor(GCD)

Great Common Divisor (GCD)

We give the definition of great common divisor of $a, b \in \mathbb{Z}$ below.

Definition B.1.3 (Great Common Divisor)**.** Let $a, b \in \mathbb{Z}$, then d is great common divisor of a, b if $d \geq e$ where e is an arbitrary common divisor of a, b, written as $d = gcd(a, b) = (a, b)$.

With definition of GCD, we know that $d \geq e$ for any common divisor e of a, b. Here we give a theorem of d and common divisors of a, b.

Theorem B.1.2. *Let $a, b \in \mathbb{Z}$, $d = gcd(a, b)$. Let e be any common divisor of a, b. Then $e|d$.*

We will illustrate the proof of this theorem when discussing *Linear Combination of GCD*.

Euclidean Algorithm

Euclidean Algorithm is the algorithm to find $gcd(a, b)$. We will illustrate the algorithm below.

Linear Combination of GCD

Here we introduce another important property of GCD.

Theorem B.1.3. *Let $a, b \in \mathbb{Z}$ and $d = gcd(a, b)$. Then $d = ax + by$ for $x, y \in \mathbb{Z}$*

This theorem can be proved by induction with Euclidean Algorithm. We will show several examples below.

Relative Prime

Definition B.1.4 (Relative Prime)**.** Let $a, b \in \mathbb{Z}$ and $d = gcd(a, b)$. Then a, b are said to be relatively prime if $d = 1$. Let $a_1, a_2, \cdots, a_k \in \mathbb{Z}$. Then $a_1, a_2, \cdots, a_k$ are said to be pairwise relatively prime if $gcd(a_i, a_j) = 1$ for all $a_i, a_j \in \{a_1, a_2, \cdots, a_k\}$.

B.1.4 Congruence

Congruence and Congruence Class

Definition B.1.5 (congruence)**.** On $\mathbb{Z}$, fix $n \in \mathbb{Z}^+$, then a is congruent to b if and only if $n|a - b$, written as

$$a \equiv b \pmod{n} \tag{B.1}$$

Given n, possible remainders are $\{0, 1, \cdots, n-1\}$.
Define the congruence class of a modulo n as below :

Definition B.1.6 (Congruence Class)**.** $[a]_n = \{a \in \mathbb{Z} : x \equiv a \pmod{n}\}$

Theorem B.1.4. *$a \equiv b \pmod{n}$ if and only if $[a]_n = [b]_n$.*

Therefore, the congruence classes modulo n are $\mathbb{Z}_n = \{[0], [1], \cdots [n-1]\}$.

Proposition B.1.1. *If $n \in \mathbb{Z}^+$ and $a, b \in \mathbb{Z}$ such that $a \equiv b \pmod{n}$, then $gcd(a, n) = gcd(b, n)$.*

Note that $gcd([a], n) = gcd(a, n)$ is well defined. Then $\mathbb{Z}_n^\times = \{[a] \in \mathbb{Z}_n : gcd([a], n] = 1\}$ and $\phi(n) = |\mathbb{Z}_n^\times|$.

Multiplicative Inverse

We first give definition of multiplicative inverse.

Definition B.1.7 (Multiplicative Inverse)**.** If there exists $a^{-1} \in \mathbb{Z}_n$ such that for $a \in \mathbb{Z}_n$,

$$a \cdot a^{-1} \equiv 1 \pmod{n}$$

Then a^{-1} is the multiplicative inverse of a modulo n.

Theorem B.1.5 (Existence of Multiplicative Inverse)**.** *$a \in \mathbb{Z}_n$ has a multiplicative inverse modulo n if and only if $gcd(a, n) = 1$.*

We can use *Extended Euclidean Algorithm* to find the multiplicative inverse of a modulo n. As we discussed above, the $gcd(a, n)$ can be written as the linear combination of a, n. Thus we can derive a theorem to find the multiplicative inverse of a modulo n.

Theorem B.1.6. *Let $gcd(a, n) = 1$ and $1 = ax + ny$ for $x, y \in \mathbb{Z}$. x is the multiplicative inverse of a modulo n i.e., $x \cdot a \equiv 1 \pmod{n}$.*

B.1.5 Chinese Remainder Theorem

Theorem B.1.7 (Chinese Remainder Theorem(CRT))**.** *Let $m_1, m_2, \cdots, m_n$ be pairwise relative prime. Then there exists $a_1, a_2, \cdots, a_n \in \mathbb{Z}$ such that:*

$$\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \\ \vdots \\ x \equiv a_n \pmod{m_n} \end{cases}$$

Solve system of congruence

CRT gives the idea of solving system of congruence. We will illustrate the idea below. Let $m_1, m_2, \cdots, m_n$ be pairwise relative prime. Let $M = \prod_{i=1}^{n} m_i$ and $\mu_i = \prod_{j \neq i} m_j$. Construct the multiplicative inverse $\mu_i \cdot \overline{\mu_i} \equiv 1 \pmod{m_i}$. The congruence system

$$\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \\ \vdots \\ x \equiv a_n \pmod{m_n} \end{cases}$$

has the solution

$$x \equiv \sum_{i=1}^{n} a_i \cdot \mu_i \cdot \overline{\mu_i} \pmod{M}$$

B.1.6 Euler's Theorem and Fermat's Little Theorem

Theorem B.1.8 (Fermat's Little Theorem)**.** *Let p be prime and $a \in \mathbb{Z}^+$, then*

$$a^{p-1} \equiv 1 \pmod{p}$$

To discuss Euler's Theorem, we first introduce the Euler's Totient Function.

Definition B.1.8 (Euler's Totient Function)**.** Let $n \in \mathbb{Z}^+$, then the Euler's Totient Function $\phi(n)$ is the number of positive divisors which are relatively prime to n.

Theorem B.1.9 (Euler's Theorem)**.** *Let $a, n \in \mathbb{Z}^+$ and $gcd(a, n) = 1$. Then*

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

B.1.7 Modular Exponent

Definition B.1.9 (Order of a modulo m)**.** Let $a, m \in \mathbb{Z}^+$ and $gcd(a, m) = 1$, then the order of a is the smallest integer k such that $a^k \equiv 1 \pmod{m}$, denoted $k = ord_m(a)$.

B.1.8 Quadratic Residue

Definition B.1.10 (Quadratic Residue)**.** Let $n \in \mathbb{Z}$. If there exists $t \in \mathbb{Z}$ such that $x^2 \equiv t \pmod{n}$ where $x \in \mathbb{Z}$, then t is a quadratic residue modulo n. Otherwise, t is a quadratic non-residue modulo n.

Theorem B.1.10 (Euler's Criterion)**.** *Let p be an odd prime and $a \in \mathbb{Z}$ such that $p \nmid a$. Then $x^2 \equiv a \pmod{p}$ is solvable or not according as :*

$$a^{\frac{p-1}{2}} \equiv 1 \pmod{p} \text{ or } a^{\frac{p-1}{2}} \equiv -1 \pmod{p}$$

Legendre Symbol

Definition B.1.11 (Legendre Symbol)**.** Let p be a prime and $a \in \mathbb{Z}$ such that $p \nmid a$. Then the Legendre Symbol

$$(\frac{a}{p}) = \begin{cases} 1 \text{ if } x^2 \equiv a \pmod{p} \\ -1 \text{ if } x^2 \not\equiv a \pmod{p} \end{cases}$$

Legendre Symbol can be used to determine if an integer is a quadratic residue. Thus we can combine Euler's Criterion and Legendre Symbol.

Theorem B.1.11 (Euler's Theorm (Legendre Symbol))**.** *Let p be an odd prime and $a \in \mathbb{Z}$ such that $p \nmid a$. Then $(\frac{a}{p}) \equiv a^{\frac{p-1}{2}} \pmod{p}$.*

We now introduce some properties of Legendre Symbol.

Theorem B.1.12 (Property of Legendre Symbol)**.** *Let $a, b \in \mathbb{Z}$ and p be an odd prime.*

1. *If $a \equiv b \pmod{p}$, then $(\frac{a}{p}) = (\frac{b}{p})$.*
2. *$(\frac{-1}{p}) = (-1)^{\frac{p-1}{2}}$.*
3. *$(\frac{ab}{p}) = (\frac{a}{p})(\frac{b}{p})$ provided that $p \nmid ab$.*

Theorem B.1.13 (Quadratic Reciprocity)**.** *Let p, q be distinct prime. Then*

$$(\frac{p}{q}) = (-1)^{\frac{p-1}{2} \cdot \frac{q-1}{2}} (\frac{q}{p})$$

B.1.9 Discrete Logarithm Problem

To discuss discrete logarithm problem, we need to give the definition of a primitive root.

Primitive Root

Definition B.1.12 (primitive root modulo m)**.** Let $a, m \in \mathbb{Z}^+$ and $gcd(a, m) = 1$. Then a is a primitive root modulo m if $a^{\phi(n)} \equiv 1 \pmod{m}$ and $ord_m(a) = \phi(n)$, then a is a primitive root modulo m.

Now we describe the discrete logarithm problem modulo m.

Definition B.1.13 (Discrete Logarithm Problem Modulo m)**.** Let $m \in \mathbb{Z}^+$ and a be a primitive root modulo m. Let $b \in \mathbb{Z}^+$ and $gcd(b, m) = 1$. The discrete logarithm problem modulo m is to find the integer k such that $a^k \equiv b \pmod{m}$, denoted $ind_a(b)$ where m is fixed.

B.2 Primality Test

Primality test is to check an odd integer $n > 1$ is composite or **possibly** prime.

B.2.1 Miller-Rabin Primality Test

We illustrate Miller-Rabin Primialiy test below. Assume that $n > 1$ be an odd integer.

Algorithm B.1 Miller-Rabin Primality Test

1: Write $2m = n - 1$ where m is odd. Choose k rounds of computation.
2: Choose a random integer a with $1 < a < n - 1$.
3: Compute $b_0 \equiv a^m \pmod{n}$.
4: If $b_0 \equiv \pm 1 \pmod{n}$, then n is possibly prime.
5: Else iteratively compute $b_i \equiv b_{i-1}^2 \pmod{n}$ for $i = 1, 2, \cdots$
6: If $b_i \equiv 1 \mod n$, then n is composite. Else if $b_2 \equiv -1 \pmod{n}$, then n is probably prime.
7: Continue until stopping or until reaching b_{k-1}. If $b_{k-1} \equiv -1 \pmod{n}$, then n is composite.

B.2.2 Solovy-Strassen Primality Test

The Solovy-Strassen Primality Test uses the property of Jacobi Symbol to prove if an integer is prime.

Theorem B.2.1 (Solovy-Strassen Primality Test)**.** *Let $(\frac{a}{n})$ be a Jacobi Symbol. If $\frac{a}{n} \equiv a^{\frac{n-1}{2}} \pmod{n}$, the n is probably a prime. Else if $\frac{a}{n} \not\equiv a^{\frac{n-1}{2}} \pmod{n}$, then n is a composite.*

B.2.3 Lucas Primality Test

B.2.4 Pockington Primality Proving

Pockington Primality Proving is based on Euler's Theorem. Assume that $gcd(a, n) = 1$, then $a^{\phi(n)} \equiv 1 \pmod{n}$. Assume that n is prime, then we know that $\phi(n) = n - 1$. Thus we can check if n is a prime by checking if $a^{n-1} \equiv 1 \pmod{n}$.

Theorem B.2.2 (Pocklington Primality Proving)**.** *Let $n \in \mathbb{Z}^+$. If $a \in \mathbb{Z}^+$ and $gcd(a, n) = 1$ such that $ord_n(a) = n - 1$. Then n is a prime.*

Appendix C

Abstract Algebra

This section introduces the Abstract Algebra knowledge which is required for cryptography including Group,Ring and Field.

C.1 Group

This section mainly introduces the knowledge about Group Theory.

C.1.1 Definition of a Group

Definition C.1.1. A binary operation $*$ on the set A is a function. $f : A \times A \to A$ where $f(a, b) = a * b$.

Define a group with binary operation $*$ as following :

Definition C.1.2. (Definition of Group) Let $*$ be the binary operation on the set G.

1. Closed under operation $*$
2. $*$ is associative
3. every element of G has an inverse
4. G has an identity element for $*$

Proposition C.1.1. *If $a *$ is a binary operation on A, then it has at most* 1 *identity element.*

Definition C.1.3 (Identity Element)**.** If $*$ is a binary operation on A with identity e, then $a \in A$ has inverse $b \in A$ if $a * b = b * a = e$.

C.1.2 Homomorphism and Isomorphism

Definition C.1.4 (Isomorphism)**.** Let G_1, G_2 be groups, a function $\phi : G_1 \to G_2$ is an isomorphism if :

1. ϕ is bijective(i.e., one-to-one and onto)
2. $\forall g, h \in G_1, \phi(g *_1 h) = \phi(g) *_2 \phi(h)$

Then we say G_1 and G_2 are isomorphic,written as $G_1 \cong G_2$. (Note that $\cong$ is an equivalence relation)

Example C.1.1. Let $2\mathbb{Z} = \{2k : k \in \mathbb{Z}\}$ with $+$, then $\mathbb{Z} \cong 2\mathbb{Z}$.

Definition C.1.5 (Homomorphism)**.** Let G_1, G_2 be groups, a function $\phi : G_1 \to G_2$ is a homomorphism if $\forall a, b \in G_1, \phi(a *_1 b) = \phi(a) *_2 \phi(b)$. (Note that an isomorphism is a bijective homomorphism)

Example C.1.2. If G_1 and G_2 are groups. Define $f : G_1 \to G_2$ by $f(x) = e_2$ (identity function) is a homomorphism.

Proposition C.1.2. *If $\phi : G_1 \to G_2$ is a group isomorphism, then*

1. $\phi(e_1) = e_2$
2. $\forall a \in G_1, \phi(a^{-1}) = \phi(a)^{-1}$

Proposition C.1.3. *If $G_1 \cong G_2$ and G_1 is abelian and G_2 is also abelian.*

C.1.3 Kernel and Image

Definition C.1.6 (Kernel)**.** If $\phi : G_1 \to G_2$ is a homomorphism, then the kernel of ϕ is : $ker(\phi) = \{a \in G_1 : \phi(a) = e_2$ where e_2 is identity of G_2. (Note that $ker(\phi)$ is a subgroup of G_1)

Example C.1.3. For $\phi : \mathbb{Z} \to \mathbb{Z}_n$,given by $\phi(a) = [a]_n$. Then $ker(\phi) = \{nk : k \in \mathbb{Z}\} = n\mathbb{Z}$.

Theorem C.1.1. *If $\phi : G_1 \to G_2$ is a homomorphism, ϕ is injective if $Ker(\phi) = \{e\}$.*

Definition C.1.7. If G is an abelian group and $m \in \mathbb{Z}$, $[m] : G \to G$, given by

$$[m](g) = g^m = \begin{cases} g * g * \cdots * g \text{ if } m > 0 \\ e \text{ if } m = 0 \\ (g^{-1})^{-m} \text{ if } m < 0 \end{cases} \tag{C.1}$$

In additive notation :

$$[m](g) = m \cdot g = \begin{cases} g + g + \cdots + g \text{ if } m > 0 \\ e \text{ if } m = 0 \\ (-g) + (-g) + \cdots + (-g) \text{ if } m < 0 \end{cases} \tag{C.2}$$

Proposition C.1.4. *If $m, n \in \mathbb{Z}^+$ and $x \in \mathbb{Z}$, $m \cdot x \equiv 0 \pmod{n}$ if and only if $gcd(m, n) \cdot x \equiv 0 \pmod{n}$*

Corollary C.1.1. *If $m, n \in \mathbb{Z}^+, d = gcd(m, n)$, then on $\mathbb{Z}_n$, $ker([m]) = ker([d])$.*

C.1.4 Cyclic Groups

Definition C.1.8 (Cyclic Groups)**.** Let $g \in G$, the cyclic group generated by g is $< g >= \{g^n : n \in \mathbb{Z}\} (= \{n \cdot g : n \in \mathbb{Z}\}$ in additive notation)

Definition C.1.9 (Generator)**.** If $G =< g >$ where $g \in G$, then g is called a generator of G.

Proposition C.1.5. *If $g \in G$, $< g >$ is a subgroup of G. Then we call $< g >$ the cyclic subgroup generated by g. If $G =< g >$ for some $g \in G$, we say G is a cyclic group.*

Example C.1.4. Let $G = \mathbb{Z}, n \in G, < n >= \{n \cdot m : m \in \mathbb{Z}\}$

Example C.1.5. In $\mathbb{Z}_4$,

$$< 0 >= \{0\}$$
$$< 1 >= \{n \cdot 1 : n \in \mathbb{Z}\} = \{0, 1, 2, 3\}$$
$$< 2 >= \{n \cdot 2 : n \in \mathbb{Z}\} = \{0, 2\}$$
$$< 3 >= \{n \cdot 3 : n \in \mathbb{Z}\} = \{0, 2, 3\}$$

C.1.5 Torsion Element and Order of Element

Definition C.1.10 (m-torsion)**.** If $g \in G$ and $m \in \mathbb{Z}^+$ and $g^m = e$ (in additive notation : $m \cdot g = e$) , then g is m-torsion. (Note that g is 1-torsion if and only if $g = e$)

Definition C.1.11 (order)**.** If $g \in G$ and $m \in \mathbb{Z}^+$ and $g^m = e$ (in additive notation : $m \cdot g = e$) , then the smallest such m is the order of g, written as $|g|$. Otherwise, we say g has infinite order i.e., $|g| = \infty$. (Note that $|g| = 1$ if and only if $g = e$)

Theorem C.1.2. *Let G be a group and $g \in G$.*

1. *If g has infinite order, then g^i for some $i \in \mathbb{Z}$ are distinct. Moreover, $\mathbb{Z} \cong < g >$ by $i \mapsto g^i$.*
2. *If g has order n, then*
   (a) *$< g >= \{g^0, g^1, \cdot g^{n-1}\}$ and this has order n.*
   (b) *If $i \in \mathbb{Z}$, then $g^i = g^{i \mod n}$.*
   (c) *$\forall i, j \in \mathbb{Z}, g^i = g^j$ if and only if $i \equiv j \pmod n$.*
   (d) *$\forall i \in \mathbb{Z}$, $g^i = e$ if and only if $n|i$*
   (e) *$\mathbb{Z}_n \cong < g >$ by $[i]_n \mapsto g^i$*

Note that : $\forall g \in G, |g| = | < g > |$

Corollary C.1.2. *If $g \in G$ and $|g| = n < \infty$, then g is $m-$torsion if and only $n|m$.*

Definition C.1.12. If G is abelian, $[m] : G \to G$ is a homomorphism and $G[m] = ker([m])$. (Note that $G[m]$ is a subgroup of G)

Definition C.1.13 ($Ord(G, m)$)**.** If G is a group, $m \in \mathbb{Z}^+$, let $Ord(G, m) = \{g \in G : |g| = m\}$.

Proposition C.1.6. *Let $n \in \mathbb{Z}^+$ and $a \in \mathbb{Z}_n$ and $d = gcd(a, n)$, then $< a >=< d >$*

Proposition C.1.7. *Let $n, m \in \mathbb{Z}^+$ and $d = gcd(n, m)$. In $\mathbb{Z}_n$:*

1. *$ker([m]) = ker([d])$*
2. *$Im([m]) = Im([d])$*
3. *$Im([m] =< m >$*
4. *$ker([d]) = Im([\frac{n}{d}])$*
5. *$< d >= \{0 \cdot d, 1 \cdot d, \cdots, (\frac{n}{d} - 1) \cdot d\}$*

Corollary C.1.3. *In* $\mathbb{Z}_n, |m| = |gcd(n,m)| = \frac{n}{gcd(n,m)}$. *In particular,* $d|n$, *then* $|d| = \frac{n}{d}$

Corollary C.1.4. *If* $n \in \mathbb{Z}^+$ *and* $a, bin\mathbb{Z}$, *then :*

$$ax \equiv b \pmod{n}$$

has solution if and only if $gcd(a,n)|b$.*(Note that means to check if* $b \in Im([a])$ *)*

Corollary C.1.5. *If* G *is a group,*$g \in G$ *with* $|g| = n < \infty$, *then* $|g^i| = \frac{n}{gcd(i,n)}$

Corollary C.1.6. *If* $d, n \in \mathbb{Z}^+$ *and* $d|n$, *then :*

$$\phi(d) = |\{a \in \mathbb{Z}_n : |a| = d\}|$$

If $m \nmid n$, $\{a \in \mathbb{Z}_n : |a| = m\} = \emptyset$
Note that : $d = |a| = \frac{n}{gcd(a,n)} \Leftrightarrow gcd(a,n) = \frac{n}{d}$

C.1.6 Cosets and Quotient Groups

Definition C.1.14 (Coset)**.** Let G be a group and $H \leq G$. Then the set $gH|g \in G$ is the left coset of H. Similarly, the set $Hg|g \in G$ is the right coset of H.

Definition C.1.15 (Normal of Groups)**.** If G is a group, N is a normal of G if N is a subgroup of G and every left coset of N is the right coset of N. Denoted as $N \trianglelefteq G$.

Definition C.1.16 (Quotient Groups)**.** if $N \trianglelefteq G$, let $G/N = \{gN|g \in G\}$ with operations $(g_1N)(g_2N) = (g_1g_2N)$. Then G/N is called a quotient group (factor group) of G.

C.1.7 Automorphism

Definition C.1.17 (Automorphism)**.** Let G be a group. The set $Aut(G) = \{\phi : G \to G|f\text{is an isomorphism}\}$ is called the set of automorphisms of G. If $\phi \in Aut(G)$, then ϕ is called an automorphism.

Definition C.1.18 (Conjugation and Inner Automorphism)**.** Let $\phi_g : G \to G$ be defined as $a \mapsto gag^{-1}$ where $a, g \in G$. Then ϕ_g is called a conjugation of g. Define the set $Inn(G) = \{\phi_g|g \in G\}$ as the set of inner automorphisms of G.

Theorem C.1.3 (The Fundamental Theorem of Homomorphsim)**.** *Let* $\phi : G \to H$ *be a group homomorphism, then* $G/Ker(f) \cong \phi(G)$ *by the isomorphism* $\psi : G/Ker(f) \to \phi(G)$ *defined by* $gKer(\phi) \mapsto \phi(g)$.

Note that *The Fundamental Theorem of Homomorphsim* also refers to The First Isomomorphism Theorm.

C.2 Ring

This section mainly introduces the knowlege about Ring Theory

C.2.1 Definition of a Ring

We first give the definition of a ring below.

Definition C.2.1 (Definition of Ring)**.** A ring R is a set of objects, called elements on which is defined two binary operation $+, \cdot$. A ring has to satisfy the following axioms.

1. R is an abelian group under $+$.
2. If $a, b \in R$, then $a \cdot b \in R$.
3. (Associative) $a \cdot (b \cdot c) = (a \cdot b) \cdot c$.
4. (Distributive) $a \cdot (b + c) = a \cdot b + a \cdot c$, $(a + b) \cdot c = a \cdot c + b \cdot c$

There are also special rings which satisfies certain axioms.

Definition C.2.2 (Communicative Ring)**.** Let R be a ring and $a, b \in R$. If $a \cdot b = b \cdot a$, then R is a commutative ring.

Definition C.2.3 (Communicative Ring)**.** Let R be a ring and there exists $1_R \in R$ such that for all $a \in R$, $a \cdot 1_R = 1_R \cdot a = a$. Then R is a ring with identity 1_R.

We know give definition on special elements in a ring.

Definition C.2.4 (Zero Divisor)**.** Let $u \in R$ and $u \neq 0_R$. If $\exists v \in R$ and $v \neq 0_R$ such that $u \cdot v = 0_R$ or $v \cdot u = 0_R$, then u is a zero divisor in R.

Definition C.2.5 (Unit)**.** Let R be a ring with identity. An element $u \in R$ is said to be a unit of R IF it has a multiplicative inverse.

C.2.2 Ring Homomorphism and Isomorphism

In Group section, we introduce group homomorphism and isomorphism. To be a ring homomorphism and isomorphism, the condition must be satisfied for both additive and multiplicative operations.

Definition C.2.6 (Isomorphism)**.** Let R_1, R_2 be rings, a function $\phi : R_1 \to R_2$ is an isomorphism if :

1. ϕ is bijective(i.e., one-to-one and onto)
2. $\forall g, h \in R_1, \phi(g +_1 h) = \phi(g) +_2 \phi(h)$
3. $\forall g, h \in R_1, \phi(g \cdot_1 h) = \phi(g) \cdot_2 \phi(h)$

Then we say R_1 and R_2 are isomorphic,written as $R_1 \cong R_2$. (Note that $\cong$ is an equivalence relation)

Example C.2.1. Let $2\mathbb{Z} = \{2k : k \in \mathbb{Z}\}$ with $+$, then $\mathbb{Z} \cong 2\mathbb{Z}$.

Definition C.2.7 (Homomorphism)**.** Let R_1, R_2 be groups, a function $\phi : R_1 \to R_2$ is a homomorphism if

1. $\forall g, h \in R_1, \phi(g +_1 h) = \phi(g) +_2 \phi(h)$
2. $\forall g, h \in R_1, \phi(g \cdot_1 h) = \phi(g) \cdot_2 \phi(h)$

(Note that an isomorphism is a bijective homomorphism)

C.2.3 Integral Domain

Definition C.2.8 (Definition of Integeral Domain)**.** An integral domain is commutative ring with 1_R and possessing no zero divisor.

C.2.4 Quotient Ring

C.3 Field

This section mainly introduces the knowlege about Field. We first give the definition of a field.

Definition C.3.1 (Definition of Field)**.** A field is an integral domain in which every non-zero element has a multiplicative inverse.

Definition C.3.2 (algebraic)**.** Let K be a field. If $F \subseteq K$ are fields and $\alpha \in K$, α is algebraic over F if $\exists f \in F[x]/\{0\}$ such that $f(\alpha) = 0$.

Definition C.3.3 (algebraic closure)**.** If K is a field, there exists a field $\overline{K}$ such that $k \subseteq \overline{K}$

1. $\forall \alpha \in \overline{K}$, α is algebraic over $\overline{K}$.
2. $\forall f \in \overline{K}[x]$ of degree greater or equal 1 factors over $\overline{K}$ as a product of linear factors.

$\overline{K}$ is a algebraic closure of K.

C.3.1 Finite Field

Here we first recall binomial formula :

$$(a+b)^n = a^n + \binom{n}{1}a^{n-1}b + \binom{n}{2}a^{n-2}b^2 + \cdots + b^n$$

Proposition C.3.1. *If p is prime, and $0 < k < p$, then $\binom{p}{k}$ is a multiple of p.*

Proposition C.3.2 (Freshman's Dream)**.** *If K is a field of characteristic p, then*

$$(a+b)^p = a^p + b^p$$

We give the definition of q_{th} Frobenius map below.

Definition C.3.4 (Frobenius Map)**.** Let K be a finite field, and $p = char(K)$. Let $q = p^n$. The map $\phi_q(x) = x^q$ is called the q_{th} Frobenius Map.

Proposition C.3.3. *If K is a finite field, and $p = char(K)$, then $\phi_p : K \to K$ is a ring homomorphism.*

Proposition C.3.4. *If $a \in \mathbb{F}_q$, then $a^q = a$.*

By the previous proposition, we can conclude that :

$$\mathbb{F}_q = \{a \in \overline{\mathbb{F}_q} : a^q = a\}$$

Theorem C.3.1. *If K is a finite field, then $char(K) = p$ for some prime p, $|K| = p^n$ for some n and K has a subfield isomorphic to $\mathbb{F}_p$. For all prime p and $n \in \mathbb{Z}^+$, $\overline{\mathbb{F}_q}$ contains a unique subfield of order p^n.*

Definition C.3.5 (Automorphism)**.** If K is a field, Automorphism of the field K is the set

$$Aut(K) = \{f : K \to K : f \text{ is a ring isomorphism}\}$$

We will illustrate some examples of Automorphsim below.

Definition C.3.6. If $K \subseteq L$ are fields,

$$Aut(L/K) = \{\sigma \in Aut(L) : \forall a \in K, \sigma(a) = a\}$$

Definition C.3.7 (Galois Automorphism)**.** If $K \subseteq L$ are fields and $K = \{a \in L : \forall \sigma \in Aut(L/K), \sigma(a) = a\}$, then we say L/K is Galois. Write $Gal(L/K)$ for $Aut(L/K)$.

Proposition C.3.5. *If K is finite or $char(K) = 0$, then $\overline{K}/K$ is Galois.*

Construction of Finite Field

C.4 Polynomial

Definition C.4.1 (Polynomial)**.** Let $c_0, c_1, \cdots, c_n \in R$ where R is a ring. Then a polynomial $f(x)$ can be written as :

$$f(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_n x^n$$

Definition C.4.2 (Associate of Polynomial)**.** $f(x)$ is an associate of $g(x)$ in $F[x]$ if and only if $f(x) = c \cdot g(x)$ for some nonzero $c \in F$.

Definition C.4.3 (Irreducible Polynomial)**.** Let F be a field. Then $p(x) \in F[x]$ is irreducible if if all of its divisors are its associates and units.

Definition C.4.4 (Polynomial Congruence)**.** Let F be a field and $f(x), g(x), p(x) \in F[x]$. Then $f(x)$ is congruent to $g(x)$ modulo $p(x)$ if $p(x)|f(x) - g(x)$, written as $f(x) \equiv g(x) \pmod{p(x)}$.

Appendix D

Elliptic Curve Basic

D.1 Curves and Space

D.1.1 Affine space and Projective Space

Definition D.1.1 (Affine Space)**.** Let K be a field, define affine n-space to be K^n as a generalization of $\mathbb{R}^n$,denoted as $\mathbb{A}^n(k) = k^n$.

Definition D.1.2 (Equivalence Relation $\sim$)**.** Let $(a, b) \neq (0, 0), a \sim b \iff (a, b) = \lambda(c, d)$ for some $\lambda \in K$. Here $\lambda(c, d) = (\lambda c, \lambda d)$ as in scalar multiplication from linear algebra. Write $[a : b]$ for the equivalence class of (a, b).

Proposition D.1.1. *Let K be a field, define $\mathbb{P}^1(K)$ to be the set of equivalence classes. Then $\mathbb{P}^1(K) = \{[1 : m] : m \in K\} \cup \{[0 : 1]\}$.*

D.1.2 Homogenization and Dehomogenization

I will illustrate the process of homogenization and dehomogenization below.

Definition D.1.3 (Homogenization)**.** Homogenization is the process to convert points in affine coordinates (x, y) to projective coordinates $[X : Y : Z]$ by replacing variables x, y with $\frac{X}{Z}, \frac{Y}{Z}$.

Example D.1.1. Homogenize $y = mx + b$. Write $\frac{Y}{Z} = m\frac{X}{Z} + b$. Then we have that $Y = mX + bZ$.

Definition D.1.4 (Dehomogenization)**.** Dehomogenization is the process to convert points in projective coordinates $[X : Y : Z]$ to affine coordinates (x, y). We say that "dehomogenize with respect to Z" to replace $\frac{X}{Z}$ with u, $\frac{Y}{Z}$ with v and set $Z = 1$. Similarly with respect to X, Y.

Example D.1.2. Dehomogenize $Y = mX + bZ$ with respect to Y. We set $u = \frac{X}{Y}, v = \frac{Z}{Y}$ and $Y = 1$. Thus $1 = mu + bv$.

D.1.3 Singular and Smooth Curve

The definition of a singular point is given below.

Definition D.1.5 (singular point)**.** Let $f(x, y) = g(x, y)$ be a curve. A point $P = (x_0, y_0)$ on the curve is singular if :

1. $\frac{\partial f}{\partial x}(x_0, y_0) = \frac{\partial g}{\partial x}(x_0, y_0)$

2. $\frac{\partial f}{\partial y}(x_0, y_0) = \frac{\partial g}{\partial y}(x_0, y_0)$

Otherwise P is non-singular (or smooth).

Definition D.1.6 (singular curve)**.** We say a curve is singular if and only if every point on the curve (including the point at infinity) is singular. Otherwise, the curve is non-singular(or smooth).

Theorem D.1.1. *To say a curve is smooth or non-singular if there are no singular point over* $\overline{K}$*.*

Note that a curve is not an elliptic curve if it has singular point.

D.2 Elliptic Curves

D.2.1 Group Law

Elliptic Curves are abelian groups under addition. I now illustrate the addition law of points on elliptic curves.

Let $E : y^2 = x^3 + Ax + B$ be the elliptic curve and $P_1 = (x_1, y_1), P_2 = (x_2, y_2)$ and $P_3 = (x_3, y_3) = P_1 + P_2$. I will show addition of P_1, P_2 under different cases.

Case 1: $x_1 \neq x_2$
In this case, we compute the slope m of the line l through P_1, P_2 by

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

Thus we have that :

$$l : y - y_1 = m(x - x_1)$$

By substitution,

$$m^2(x - x_1)^2 + 2m(x - x_1)y_1 + y_1^2 = y^2 = x^3 + Ax + B$$

By the equation above, we have that

$$x^3 - m^2 - x^2 + \cdots = 0$$

Note that $(x - a)(x - b)(x - c) = x^3 - (a + b + c)x^2 + \cdots$.
So $m^2 = x_1 + x_2 + x_3$ and $x_3 = m^2 - x_1 - x_2$.
Now we obtain $y_3' = m(x_3 - x_1) + y_1$.
By negating with respect to x-axis, we get P_3 where

$$x_3 = m^2 - x_1 - x_2,\ y_3 = m(x_1 - x_3) - y_1$$

.

Case 2: $x_1 = x_2$ and $y_1 = -y_2$
In this case, the line through P_1, P_2 is vertical. We assume that the line reach the point at infinity. Thus

$$P_1 + P_2 = \mathcal{O}$$

Case 3: $x_1 = x_2$ and $y_1 = y_2 \neq 0$
In this case, we take implicit differentiation of the equation to obtain the slope m.

$$2y \cdot \frac{dy}{dx} = 3x^2 + A$$

$$m = \frac{dy}{dx}(x_1, y_1) = \frac{3x_1^2 + A}{2y_1}$$

Thus we have that :

$$l : y - y_1 = m(x - x_1)$$

Similarly by substitution, we have that :

$$x_3 = m^2 - 2x_1,\ y_3 = m(x_1 - x_3) - y_1$$

Case 4: $P_1 = \mathscr{O}$
Define the point at infinity $\mathscr{O}$ as the identity in elliptic curves. Thus for all P on the elliptic curve E :

$$\mathscr{O} + P = P + \mathscr{O} = P$$

D.2.2 Hasse Bound

Theorem D.2.1 (Hasse's Theorem)**.** *Let elliptic curve E over finite field $\mathbb{F}_q$, say $E(\mathbb{F}_q)$. Then the order of $E(\mathbb{F}_q)$, $|E(\mathbb{F}_q)| = q + 1 - a_q$ where $|a_q| \leq 2\sqrt{q}$*

Derivation of Hasse's Theorem

This section includes knowledge of Isogeny (Section 5.2.4) and Weil Pairing (Section 5.2.5). Readers are expected to learn basic concepts of Isogeny and Weil Pairing first.

I first introduce some knowledge of characteristic polynomials. Denote the characteristic polynomial of $n \times n$ matrix A as below :

$$f(x) = det(x \cdot I - A) = x^n - Tr(A) \cdot x^{n-1} + \cdots + (-1)^n \cdot det(A)$$

Theorem D.2.2 (Caley-Hamilton)**.** *If $f(x)$ is the characteristic polynomial of A, then $f(A) = 0$*

Given elliptic curve E over K, let $\alpha \in End(E)$. (See section "Isogeny and Endmorphism" for details). If $P \in E[n]$ and $n \in \mathbb{Z}^+$, then $nP = \mathscr{O}$ implies that $\alpha(nP) = n \cdot \alpha(P) = \mathscr{O}$. Thus we have that $\alpha(P) \in E[n]$

Restrict α to $\alpha|_{E[n]} : E[n] \to E[n]$ and α is a homomorphism. So we got that :

$$\alpha_{(n)} = \alpha|_{E[n]} \in End(E[n]) \cong End(Z_n \times Z_n) \cong M_2(\mathbb{Z}_n)$$

Consider that for $a \in \overline{\mathbb{F}_p}$, $a \in \mathbb{F}_p$ if and only if $a^p = a$. Thus for elliptic curve E over $\mathbb{F}_p$, the Frobenius map ϕ_p , $Q \in E(\overline{\mathbb{F}_p})$, $Q \in E(\mathbb{F}_p)$ if and only if $\phi_p(Q) = Q$. Thus we have that $Q - \phi_p(Q) = \mathscr{O}$, which implies that $Q \in Ker(1 - \phi_p)$. So $E(\mathbb{F}_p) = Ker(1 - \phi_p)$. Thus we have that if $1 - \phi_p$ is separable, then

$$|E(\mathbb{F}_p)| = |Ker(1 - \phi_p)| = deg(1 - \phi_p)$$

.

Now consider that Weil Pairing (See section "Weil Pairing" for details) connects characteristic polynomials and frobenius map of elliptic curves over finite field :

$$\text{If } p \nmid n, deg(1-\phi_p) \equiv det(1-\phi_p)_{(n)} \pmod{n}$$

Note that for matrix $A \in M_2(\mathbb{F}_p), det(I-A) = det(A) - Tr(A) + 1$. So one can show $\exists a_p \in \mathbb{Z}, a_p \equiv Tr((\phi_p)_{(n)}) \pmod{n}$ such that

$$\phi_p^2 - a_p \cdot \phi_p + deg(\phi_p) = 0$$

Using that the $deg(\alpha) \geq 0$, one can show if $x^2 - a_p \cdot x + p$ is ever negative $\forall \alpha \in \mathbb{Q}$, then $\alpha \in \mathbb{R}$ such that $a_p^2 - 4p \leq 0$, which is $|a_p| \leq 2\sqrt{p}$. Thus I have showed the Hasse's Theorem :

$$|E(\mathbb{F}_p)| = deg(1-\phi_p) = p + 1 - a_p \text{ for } |a_p| \leq 2\sqrt{p}$$

Also,

$$E(\mathbb{F}_{p^2}) = Ker(1-\phi_{p^2}) = Ker(1-(\phi_p)^2)$$

D.2.3 j-invariant

I will give the formula to compute j-invariant of an elliptic curve in Short Weierstrass Form below.

Definition D.2.1 (j-invariant)**.** Define the j-invariant of elliptic curve $E : y^2 = x^3 + Ax + B$ as :

$$j = j(E) = 1728\frac{4A^3}{4A^3 + 27B^2}$$

Here I also want to mention several theorems involving j-invariant.

Theorem D.2.3 (transformation of elliptic curves)**.** *Let $E_1 : y_1^2 = x_1^3 + Ax_1 + B_1$ and $E_2 : y_2^2 = x_2^3 + Ax_2 + B_2$ be elliptic curves with $j_1 = j(E_1)$ and $j_2 = j(E_2)$. If $j_1 = j_2$,then there exists $\mu \neq 0 \in \overline{K}$ (algebraic closure of K) such that :*

$$A_2 = \mu^4 A_1, B_2 = \mu^6 B_1$$

Also,

$$x_2 = \mu^2 x_1, y_2 = \mu^3 y_1$$

Theorem D.2.4 (Isomorphic elliptic curves and j-invariant)**.** *Let $E_1 : y_1^2 = x_1^3 + Ax_1 + B_1$ and $E_2 : y_2^2 = x_2^3 + Ax_2 + B_2$ be elliptic curves with $j_1 = j(E_1)$ and $j_2 = j(E_2)$. Then $E_1 \cong E_2$ if and only if $j_1 = j_2$.*

D.2.4 Isogeny and Endmorphism

A generalized definition of isogeny is given below.

Definition D.2.2 (Isogeny)**.** Let E_1, E_2 be elliptic curves over field K.

$$\phi : E_1 \to E_2$$

is an isogeny if :

1. $\phi(x,y) = (\alpha(x,y), \beta(x,y))$ where α, β are rational functions.

2. $\forall(x,y) \in E_1(\overline{K}), \phi(x,y) \in E_2(\overline{K})$.

3. $\phi(\mathscr{O}_1) = \mathscr{O}_2$

Thus we can prove that an isogeny $\phi : E_1 \to E_2$ is a group homomorphism.

Proposition D.2.1. *If K is field, then $\overline{K}$ is infinite.*

Some special examples of isogeny are given below.

Example D.2.1 (Zero Isogeny(Trivial Isogeny)). $\forall P \in E_1, \phi(P) = \mathscr{O}_2$. Then ϕ is a zero isogeny (or trivial isogeny).

Here is an example of Endmorphism which is a special isogeny and the definition of Endmorphism will be given later.

Example D.2.2 (Endmorphism). For any elliptic curve E and $m \in \mathbb{Z}^+$, $[m] : E \to E$ is an Endmorphism.

Definition D.2.3 (Endmorphism)**.** Let E be elliptic curve over field K. Then Endmorphism of E is the set :

$$End(E) = \{\phi : E \to E : \phi \text{ is an isogeny } \}$$

Consider that an isogeny $\phi(x,y) = (\alpha(x,y), \beta(x,y))$ where α, β are rational functions. With steps of algebra, we can rewrite an isogeny in the form

$$\phi(x,y) = (\frac{p(x)}{q(x)}, \frac{s(x)}{t(x)} \cdot y)$$

where $p(x), q(x), s(x), t(x)$ are pairwise relatively prime.

Define the degree of an isogeny ϕ as below.

Definition D.2.4. If $\phi : E_1 \to E_2$ is an isogeny, then $deg(\phi) = \begin{cases} 0 \text{ ,if } \phi = \mathscr{O} \\ max\{p(x), q(x)\} \text{ ,otherwise} \end{cases}$

Definition D.2.5 (Separability of an Isogeny)**.** An isogeny $\phi : E_1 \to E_2$ is separable if

$$\frac{d}{dx}(\frac{p(x)}{q(x)}) \neq 0$$

We can derive a theorem regarding the separability of an Isogeny to save the time for doing derivation of rational polynomial.

Theorem D.2.5 (Separability of an Isogeny)**.** *An isogeny $\phi : E_1 \to E_2$ is separable if and only if $\frac{d}{dx}(p(x)) \neq 0$ or $\frac{d}{dx}(q(x)) \neq 0$.*

Proposition D.2.2. *If $\phi : E_1 \to E_2$ is a non-zero isogeny, then :*

1. *If ϕ is separable, then $|ker(\phi)| = deg(\phi)$.*

2. *If ϕ is not separable, then $|ker(\phi)| < deg(\phi)$*

Proposition D.2.3. *If $\phi : E_1 \to E_2$ is a non-trivial isogeny, then ϕ is onto for points in $\overline{K}$*

Corollary D.2.1. *If $m \in \mathbb{Z}^+$ and $char(k) \nmid m$, then*

$$|E(\overline{K})[m]| = |ker(m)| = m^2$$

If $char(k)|m$, then

$$|E(\overline{K})[m]| = |ker(m)| < m^2$$

Corollary D.2.2. *Let K be a finite field. Then $E(\overline{K})$ is cyclic or the product of 2 cyclic groups.*

D.2.5 The Weil Pairing

Consider that if K is a finite field or $char(K) = 0$, and $n \in \mathbb{Z}^+$ with $char(K) \nmid n$, then

$$E[n] = E(\overline{K}) \cong \mathbb{Z}_n \times \mathbb{Z}_n$$

Define the basis for $E[n]$.

Definition D.2.6 (Basis)**.** If $P, Q \in E[n]$ such that every $R \in E[n]$ can be written uniquely as $aP + bQ$ with $a, b \in \mathbb{Z}_n$, then we say P, Q are basis for $E[n]$.

Note that if $\phi : \mathbb{Z}_n \times \mathbb{Z}_n \cong E[n]$ is an isomorphism, $P = \phi(1, 0), Q = \phi(0, 1)$, then P, Q form a basis. Moreover, if we have a basis, we can construct an isomorphism ϕ from above.

Define the Weil Pairing below.

Definition D.2.7 (the Weil Pairing)**.** Let E be an elliptic curve over K and $n \in \mathbb{Z}^+$ such that $char(K) \nmid n$,then there exists a function :

$$e_n : E[n] \times E[n] \to \mu_n$$

where

$$\mu_n = \mu_n(\overline{K}) = \{a \in \overline{K} : a^n = 1\}$$

such that :

1. (Bilinearity) $\forall P, Q, R \in E[n]$,

$$e_n(P + Q, R) = e_n(P, R) \cdot e_n(Q, R)$$

$$e_n(P, Q + R) = e_n(P, Q) \cdot e_n(P, R)$$

2. (Non-degeneracy) If $P \in E[n]$ and $\forall Q \in E[n]$, $e_n(P, Q) = 1$,then $P = \mathscr{O}$. Also if $\forall P \in E[n]$, $e_n(P, Q) = 1$, then $Q = \mathscr{O}$.

3. (Alternating) $\forall P \in E[n], e_n(P, P) = 1$.

4. (Skew Symmetric) $\forall P, Q \in E[n], e_n(P, Q) = e_n(Q, P)^{-1}$

5. (Galois Invariant) $\forall P, Q \in E[n]$ and $\forall \sigma \in Gal(\overline{K}/K)$,

$$e_n(\sigma(P), \sigma(Q)) = \sigma(e_n(P, Q))$$

6. If $\phi : E \to E$ is an isogeny,then $\forall P, Q \in E[n]$,

$$e_n(\phi(P), \phi(Q)) = e_n(P, Q)^{deg(\phi)}$$

D.3 Other Applications of Elliptic Curves

D.3.1 Elliptic Curve Primality Proving

Theorem D.3.1 (Elliptic Curve Primality Proving)**.** *To prove p is prime, find an elliptic curve E over $\mathbb{Z}_p$ and points $P_i \in E(\mathbb{Z}_p)$. There exist distinct primes l_i such that $[l_i] \cdot P = \mathscr{O}$. And the product of l_i*

$$\prod_{i=1}^{k} l_i > (\sqrt[4]{p} + 1)^2$$

Then p is a prime.

D.3.2 Elliptic Curves Factorization

I will illustrate factorization using elliptic curves. Assume that we need to factor $n = pq$ where p, q are distinct primes. Compared with $p-1$ factorization method, Elliptic Curve Factorization allows running multiple elliptic curves in parallel, which improve the efficiency. Also, one can construct the primes p, q such that $p-1$ factorization failed.(See $p-1$ factorization section for details) But one can not avoid all possible elliptic curves.

Algorithm D.1 Elliptic Curve Factorization

1: Choose elliptic curve E over $\mathbb{Z}_n$, say $E(\mathbb{Z}_n)$. Point $P \in E(\mathbb{Z}_n)$. Upper bound $\beta \in \mathbb{Z}^+$
2: Compute $(j!)P$ for $j = 2, 3, \cdots, \beta$.
3: Let $m = \frac{y_i}{x_i}$ be the slope when computing multiple of P.Compute $d = gcd(x_i, n)$
4: If $1 < d < n$, then d is a non-trivial factor of n. Else proceed to next j.
5: If $(j!)P = \mathcal{O}$ for some j or $(j!)P$ does not produce a non-trivial factor of n for $j = 2, 3, \cdots, \beta$, the factorization failed. Try another elliptic curve.

The correctness of Elliptic Curve Factorization is justified below. Consider that the multiplicative inverse of x_i modulo n is required when computing the slope. The multiplicative inverse of x_i modulo n exists if and only if $gcd(x_i, n) = 1$. If $1 < gcd(x_i, n) < n$, then we have a non-trivial factor of n, then the factorization succeeds.

Appendix E

Information Theory Basic

E.1 Information Entropy

Information Entropy(also called Shannon Entropy) is the average rate at which information is produced by a stochastic source of data. In Cryptoanalysis, entropy is used as a measurement of unpredicatability of a key.

Definition E.1.1 (Entropy)**.** Let X be a random variable and $\chi = \{x_1, x_2, \cdots, x_n\}$ be the set of outcomes. The entropy of variable X is defined as

$$H(X) = -\sum_{x \in \chi} p(x) log_2 p(x)$$

Bibliography

[1] Rivest, Ronald L., Adi Shamir, and Leonard Adleman. "A method for obtaining digital signatures and public-key cryptosystems." Communications of the ACM 21.2 (1978): 120-126.

[2] ElGamal, Taher. "A public key cryptosystem and a signature scheme based on discrete logarithms." IEEE transactions on information theory 31.4 (1985): 469-472.

[3] Diffie, Whitfield, and Martin Hellman. "New directions in cryptography." IEEE transactions on Information Theory 22.6 (1976): 644-654.

[4] Massey, James L., and Jimmy K. Omura. "Method and apparatus for maintaining the privacy of digital messages conveyed by public transmission." U.S. Patent No. 4,567,600. 28 Jan. 1986.

[5] Shor, Peter W. "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer." SIAM review 41.2 (1999): 303-332.

[6] Gordon, Daniel M. "Discrete Logarithms in GF(P) Using the Number Field Sieve." SIAM Journal on Discrete Mathematics 6.1 (1993): 124-138.

[7] Chor, Benny, and Ronald L. Rivest. "A knapsack-type public key cryptosystem based on arithmetic in finite fields." IEEE Transactions on Information Theory 34.5 (1988): 901-909.

[8] Washington, Lawrence C. Elliptic curves: number theory and cryptography. Chapman and Hall/CRC, 2003.

[9] Pollard, John M. "Theorems on factorization and primality testing." Mathematical Proceedings of the Cambridge Philosophical Society. Vol. 76. No. 3. Cambridge University Press, 1974.

[10] Micciancio, Daniele. "Lattice-based cryptography." Encyclopedia of Cryptography and Security (2011): 713-715.

[11] O. Goldreich, S. Goldwasser, and S. Halevi. Public-key cryptosystems from lattice reduction problems. In Advances in cryptology, volume 1294 of Lecture Notes in Comput. Sci., pages 112–131. Springer, 1997.

[12] Gentry, Craig. "Fully homomorphic encryption using ideal lattices." Stoc. Vol. 9. No. 2009. 2009.

[13] Yao, Andrew Chi-Chih. "Protocols for secure computations." FOCS. Vol. 82. 1982.

[14] Shamir, Adi. "How to share a secret." Communications of the ACM 22.11 (1979): 612-613.

[15] Rivest, Ronald L., Adi Shamir, and Yael Tauman. "How to leak a secret." International Conference on the Theory and Application of Cryptology and Information Security. Springer, Berlin, Heidelberg, 2001.

[16] Nakamoto, Satoshi. Bitcoin: A peer-to-peer electronic cash system. Manubot, 2019.

[17] Heninger, Nadia, et al. "Mining your Ps and Qs: Detection of widespread weak keys in network devices." Presented as part of the 21st USENIX Security Symposium (USENIX Security 12). 2012.