PUBLIC KEY CRYPTOGRAPHY

RISHABH VENKATARAMANI

1. Application of Number Theory

Cryptography is the field of study of the methods of secure communication. This security is forged by *encrypting* information via different mechanisms.

1.1. **Introduction.** A **cryptosystem** is a series of algorithms to implement a particular security service for secret communication over public channels in the **crypto group**, the group of people who are attempting to communicate in secret. A **public channel** is just any communication medium that can be used to transmit and receive messages. For example, the internet can be thought of as a public channel.

Now suppose Alice wants to send Bob a message, but doesn't want Claire to know what she is sending. She must secure her message in a a **crypto cell**, a unit of the cryptosystem where Alice and Bob can communicate. Alice secures the message through an algorithm, called **encryption**, and Bob uses the same algorithm in reverse to discover the original message, known as **decryption**.

1.2. **Cæsar Cipher.** A simple method of encrypting a message is through the Cæsar cipher which shifts the letters of each word in the message by the same amount in the alphabet.

Example. The ciphered message WKLU LU DQ HQFUBSWHG THVVLJH can be shifted 3 letters backward to get the original message of THIS IS AN ENCRYPTED MESSAGE.

The problem with the Cæsar cipher is that it is very easy to decipher. There are exactly 26 different ciphers, coming from the 26 letters in the alphabet. Thus, it would be relatively easy to decipher the message, it is a matter of trial and error before revealing the original message.

1.3. **Substitution.** Another form of encryption is the Substitution cipher, where every letter is replaced by another letter in the alphabet in the encrypted message. This exponentially increases the security, as there are now 26! different ciphers. This is an example of a **monoalphabetic** cipher.

2. The Rivest-Shamir-Adleman (RSA) Cryptosystem

Definition 2.1 (RSA)**.** An asymmetric encryption algorithm that uses a key pair to mathematically encrypt and decrypt data.

In order to understand how RSA works, we must acknowledge that it is difficult to factor the product of large primes, as this is the core concept RSA is built upon. RSA requires both a public and private key. The public key, represented by the integers n and e can be known by everyone and is used for encrypting messages. By using the private key, represented by

Date: December 11, 2023.

the integer d we can decrypt the messages encrypted by the public key in a short span of time. RSA attempts to find large e, d and n so that

$$(m^e)^d \equiv m \pmod{n}. \tag{2.1}$$

2.1. **Key Generation.**

Definition 2.2 (Carmichael's Function)**.** $\lambda(n)$ is the smallest positive integer m such that $a^m \equiv 1 \pmod{n}$ where a and n are relatively prime.

Suppose we have two large primes p and q. First, calculate $n = pq$ and $\lambda(n)$. Note that since p and q are coprime, $\lambda(n) = \text{lcm}(\lambda(p), \lambda(q)) = \text{lcm}(p-1, q-1)$. $\text{lcm}(p-1, q-1)$ can be easily calculated by using the identity $ab = (\text{lcm}(a,b))(\text{gcd}(a,b))$ or through the *Euclidean Algorithm*. The value of $\lambda(n)$ is not disclosed.

There must be some integer e such that $2 < e < \lambda(n)$ and $\text{gcd}(e, \lambda(n)) = 1$. This e is part of the public key. The most common value for e is $2^{16} + 1$ in the name of efficiency. The smallest possible value while still maintaining speed would be $e = 3$.

Definition 2.3 (Bit Length)**.** The bit length of a number, $l(n)$ is the number of binary digits (also known as bits) of the number. It defined to be

$$l(n) = \lceil log_2 n + 1 \rceil. \tag{2.2}$$

Definition 2.4 (Hamming Weight)**.** The *Hamming weight* it is the number of non-zero digits of a number. Alternatively, is the sum of the digits of the binary representation of a number.

For more efficient encryption, values of e that have a small bit length and lower Hamming weight are used. The next step is determining d, which is the modular multiplicative inverse of e, or in other words

$$d \equiv e^{-1} \pmod{\lambda(n)}. \tag{2.3}$$

The value d is usually determined through the extended Euclidean algorithm and is kept as part of the private key. Note that p, q, and $\lambda(n)$ must also be kept secret in order to preserve the security of d.

Remark 2.5. The Euler totient function $\phi(n) = (p-1)(q-1)$ could also be used instead of the Carmichael function $\lambda(n)$ to calculate d (this was used in the original RSA paper). The modular congruence remains true because $\phi(n)$ is always divisible by $\lambda(n)$.

Suppose that Bob wants to send information to Alice. If they decide to use RSA, Bob must know Alice's public key to encrypt the message, and Alice must use her private key to decrypt the message.

To enable Bob to send his encrypted messages, Alice transmits her public key (n, e) to Bob via a reliable, but not necessarily secret, route. Alice's private key (d) is never distributed.

Example. Alice wants to send Bob an encrypted message. Suppose she chooses 2 primes $p = 43$ and $q = 67$ for the crypto cell:

(1) Compute $n = pq = 2881$
(2) Compute $\lambda(2881) = \text{lcm}(42, 66) = 462$
(3) Set e to an arbitrary value such as 19 such that $2 < 19 < 462$ and $\text{gcd}(19, 462) = 1$.
(4) Compute $d \equiv 19^{-1} \equiv 73 \pmod{462}$ so $d = 73$

Therefore, the public key will be $(n = 2881, e = 19)$ and the private key is $(n = 2881, d = 73)$.

Now suppose Alice wants to send Bob the encrypted value $m = 45$. To encrypt this value using our new function, she would send

$$c = 45^{19} \pmod{2881} = 161$$

to Bob. Bob having received 161 needs to know the original number:

$$m = 161^{73} \pmod{2881} = 45.$$

Remark 2.6. In real-life situations, much larger primes will be used to

2.2. **Signing.** Suppose Bob supposedly receives an encrypted message from Alice. He has no way of knowing if the message is truly from Alice, or someone else since anyone can access the public key. To verify that Alice is truly the one sending the message, she must *sign* the message by using a **hash algorithm**. This confirms that Alice is truly the individual sending the message.

3. The Diffie-Hellman Key Exchange and ElGamal Encryption

3.1. **The Diffie-Hellamn Key Exchange.** The Diffie-Hellamn key exchange protocol that allows two parties to to privately share information over an insecure channel.The purpose is to generate a key through repeated exchange of information. As a result, this key will be created from several different pieces.

3.1.1. *Key Generation.* Suppose Alice and Bob have some prime p. Both independently, Alice chooses some number a and Bob chooses some number b. Alice now computes $g^a \pmod p$ and Bob computes $g^b \pmod p$ where g is the primitive root of the multiplicative group $F_p^\times$. Alice sends $g^a \pmod p$ directly to Bob, while Bob publicly sends $g^b \pmod p$. Both can now calculate $(g^a)^b \equiv g^{ab} \pmod p$, which is the key.

Example. Suppose Alice wants to generate a key with Bob. Let $p = 31$. The smallest primitive root of 31 is 3, therefore $g = 3$. Now Alice chooses $a = 22$. She computes $3^22 \pmod{31}$, which is 14. Bob calculates $3^b \pmod{31}$ and gets 24. Therefore, the key is 28: $(3^b)^a \equiv 24^a \equiv 28 \pmod{31}$. For reference, Bob chose $b = 13$.

Remark 3.1. In real-world scenarios, p will be must larger to improve security. If Claire wanted to find out the key, she has to find $g^n \pmod p$, where $0 \leq n \leq p - 2$.

3.2. **The Discrete Logarithm Problem.** Over $\mathbb{Z}$, if we are given g and g^n, we can easily find n through a logarithm. Now suppose we want to work over $\mathbb{F}_p$ making it significantly more difficult to find n. If g and g^n are elements of $F_p^\times$, n is called the *discrete logarithm* to base g of n. Generalizing, if we have a cyclic group G with generator g n is the *discrete logarithm* to base g of n.

Definition 3.2 (Discrete Logarithm Problem)**.** Given a cyclic group G, a generator g, and some $x \in G$, find n such that $x = g^n$.

There is a similar problem that arises in the Diffie-Hellman key exchange protocol:

Definition 3.3 (Diffie-Hellman Problem)**.** Given a cyclic group G, a generator g, and both $g^a, g^b \in G$, compute g^{ab}.

It is mostly believed that there is no such algorithm that solves either of these problems, and no such algorithm as been found or proposed.

3.3. **ElGamal Encryption.** ElGamal encryption is a popular method of constructing a cryptosystem with the Diffie-Hellman key exchange.

Suppose Bob wants to send a message to Alice. Alice will have to first generate a key. She choose some prime p, generator g of $F_p^\times$, and $0 \leq a \leq p-2$. She then computes $g^a \pmod p$. Let us denote that value s. s, g, and p become part of the public key, while a is kept private. Bob chooses some $0 \leq b \leq p-2$, and computes $s^b \pmod p$ which we will denote r. The message he wants to send must be converted into some element of $\mathbb{F}_p^\times$

4. Elliptic Curve Cryptography

Definition 4.1 (Elliptic Curve)**.** An elliptic curve over a finite field is of the (Weierstrass normal) form $y^2 = x^3 + ax + b$, given $4a^3 + 27b^2 \neq 0$ and an additional point at infinity denoted ∞.

Elliptic curves are powerful tools in cryptography because there is an Abelian Group on the elliptic curve with the point at infinity as the identity element and the points of the elliptic curve as the elements of the group. The point at infinity lies on all vertical lines (of the form x = k) and no non-vertical lines.

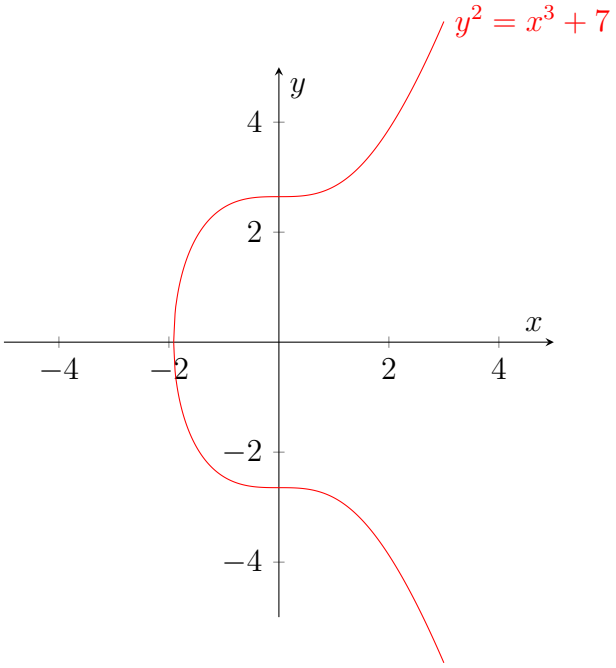

Figure 1. Elliptic curve of the form $y^2 = f(x)$

Suppose we have two points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ on the elliptic curve $y^2 = x^3 + ax + b$. Suppose we want to add these two points $P + Q$. The slope of the line connecting the two points would be

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

and intercept

$$c = y_1 - mx_1.$$

This secant line intersects the elliptic curve at another point (x_2, y_3) which must satisfy

$$y^2 = x^3 + ax + b$$

and

$$y = mx + c.$$

Plugging in $y = mx + c$ into $y^2 = x^3 + ax + b$, we get $(mx + c)^2 = x^3 + ax + b$. Expanding, rearranging and using **Vieta's formulas**, we find that $x^3 = m^2 - x_1 - x_2$ and $y_3 = mx_3 + c$. The sum $P + Q$ is the reflection of (x_3, y_3) over the x-axis: $(x_3, -y_3)$.

Using the same technique, we can find a formula for the doubling of a point P (adding it to itself). Since in this case $x_1 = x_2$ and $y_1 = y_2$, we must use calculus to find that the slope of the tangent line at P is

$$m = \frac{3x_1^2 + a}{2y_1}. \tag{4.1}$$

Furthermore, we can also find that the x-coordinate of the point $2P$ is

$$x_3 = \frac{x^4 - 2ax^2 - 8bx + a^2}{4(x^3 + ax + b)} \tag{4.2}$$

and the y-coordinate is one of the square roots of x_3.

Remark 4.2. Adding two integer points need not produce another integer point.

4.1. **Elliptic Curves over Finite Fields.** In the realm of cryptography, what interests us is the elliptic curves over finite fields, primarily $\mathbb{F}_p$ denoted $E(\mathbb{F}_p)$ ($\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$. Note that the points of $E(\mathbb{F}_p)$ form a group. Note that we still have the condition that $4a^3 + 27b^2 \neq 0$, which is replaced by the congruence

$$4a^3 + 27b^2 \not\equiv 0 \pmod{p}$$

since it is over a finite field. It is relatively simple to count the number of integer points over finite fields, since we can just plug in a finite number of values. Easier is having the **SageMath** software do it for us. In $E(\mathbb{F}_p)$, we should have a maximum of p points and another point at ∞. But, further analysis leads to a more precise bound.

Theorem 4.3 (Hasse-Weil Bound)**.** *Let E be an elliptic curve over $\mathbb{F}_p$ and let $\#E(\mathbb{F}_p)$ be the number of points in $E(\mathbb{F}_p)$. Then*

$$|\#E(\mathbb{F}_p) - (p + 1)| \leq 2\sqrt{p}. \tag{4.3}$$

It turns out that sometimes $E(\mathbb{F}_p)$ will be a cyclic group of order n. This makes E very useful for elliptic curve versions of some protocols.

Unlike prior protocols, elliptic curves had an assumption that finding the discrete logarithm of an elliptic curve element from a base point is impractical. This became known as the "elliptic curve discrete logarithm problem" and threatened the security of elliptic curve cryptography. Elliptic curve cryptography depends on the ability to compute a point multiplication on the elliptic curve and the difficulty of seeking the multiplicand. This results in smaller key sizes, as less storage and transmission can provide an equal level of security.

Euler Circle, Folsom, CA 95630
Email address: rven2308@gmail.com