INTRODUCTION TO ELLIPTIC CURVE CRYPTOGRAPHY

OLGA SHEVCHUK

Abstract. In this paper, the mathematics behind the most famous cryptographic systems is introduced. These systems are compared in terms of security, efficiency and difficulty of implementation. Emphasis is given to elliptic curve cryptography methods which make use of more advanced mathematical concepts.

Contents

1. Introduction

Until the 1970's, the encryption process was rather complicated and time-consuming because it required the two parties to meet in order to set up a shared secret key for secure communication. This was the idea behind symmetric ciphers which formed the basis of private cryptosystems. That did not cause many problems when the users of cryptography were mostly comprised of small groups of individuals that shared the system of keys which they distributed inside their military or diplomatic

Date: August 16th 2020.

organizations. The rise of computer network communication changed the average user of cryptography, and the need to make frequent transactions with different parties made private cryptography obsolete. A new kind of cryptography emerged, called asymmetric, or public - key cryptography, and involved the lengthy process of establishing the common shared key without the need to meet.

2. Public-key Cryptography Systems Overview

Diffie-Hellman was one of the first public-key cryptosystems invented. Unlike traditional private (symmetric) cryptosystems, public cryptosystems rely on one-way (trapdoor) functions, functions that are not difficult to be computed but the inverse of which takes exponentially more time to derive without the decryption key. The key required to encrypt messages can be made publicly available since it is computationally impossible to decrypt a message using encryption key only. Diffie-Hellman key exchange is a hybrid cryptosystem which combines the ideas of public-key cryptography along with a symmetric cryptosystem to transmit hidden messages. Even though public-key cryptosystems are more convenient, the explanation behind the use of the mentioned approach is that asymmetric cryptosystems are based on difficult mathematical computations and thus may be much more inefficient than symmetric ones.

2.1. **Preliminaries.**

Before introducing the Diffie - Hellman key exchange, let's start with some concepts which will justify the use of this cryptographic method.

Definition 2.1. (Order of an element in a group). Let G be a finite abelian group, written multiplicatively. Let a be any element in G. The order of a is the order (number of elements) of the subgroup generated by a, denoted by $\langle a \rangle$ which consists of all powers of a. In other words, the order of a is the minimum value of i where $i > 0$ such that $a^i = 1$.

Definition 2.2. Let $\phi(n)$ denote the Euler totient function which counts the number of integers from 1 to n (inclusive) which are coprime to n.

Definition 2.3. An element $g \in F_q^*$ is called a generator of the group F_q^*, written multiplicatively, if for every $a \in F_q^*$, we have $g^k = a$ for some integer k. In other words, the powers of g produce all elements in F_q^*. If $q = p$ where p is a prime, a generator is called a primitive root modulo p.

The following theorem establishes the existence of at least one generator in every finite field, the fact that justifies the use of those in the Diffie-Hellman key exchange.

Theorem 2.4. *Every finite field F_q has a primitive element which is the generator of the multiplicatively written group of the field.*

Proof. To show that there exists a generator in F_q^*, we need to prove the existence of an element of order $q-1$. Let a be an element of order r in the multiplicatively written group F_q^*. Let $\langle a \rangle$ denote a cyclic group generated by a:

$$\langle a \rangle = \{1, a, a^2, ..., a^{r-1}\}$$

Every element in $\langle a \rangle$ is of the form a^k for some k. So $(a^k)^r = (a^r)^k = 1$ since $a^r = 1$. Therefore, a^k has order dividing r. It follows that a^k is a root of $X^r - 1$, a polynomial over F_q^*. Then $\langle a \rangle$ forms the subset of roots of $X^r - 1$. Since the order of $\langle a \rangle$ is r and $X^r - 1$ has r roots, the number of elements in $\langle a \rangle$ equals the

number of roots of $X^r - 1$. From the properties established before, the elements of $\langle a \rangle$ are the roots of $X^r - 1$. We know that a cyclic group of order n, $\mathbb{Z}/n\mathbb{Z}$ has $\phi(n)$ generators where $\phi(n)$ is the Euler totient function. It follows that the generators correspond to the integers which are coprime to n. Then $\langle a \rangle$ has $\phi(r)$ generators or elements of order r. Let $R = \{r_1, ..., r_m\}$ denote the set of the orders of the elements in F_q^*. There are $\phi(r_i)$ elements of order r_i for every i. Since F_q^* has order $q-1$, it follows from Lagrange's theorem that $r_i|(q-1)$ for all i. Then $q - 1 = |F_q^*| = \sum_{r_i \in R} \phi(r_i)$. Let S be the set of all divisors of $q-1$. In other words, $S = \{r : r|(q-1)\}$. Then the Euler totient function satisfies the following property:

$$\sum_{r \in S} \phi(r) = q - 1.$$

R is a subset of S because $r_i|(q-1)$ for all i. Then $S = R \cup (S \setminus R)$. We have that $\sum_{r \in S} \phi(r) = \sum_{r \in R} \phi(r) + \sum_{r \in S \setminus R} \phi(r)$. Since $\sum_{r \in S} \phi(r) = \sum_{r \in R} \phi(r) = q-1$, $\sum_{r \in S \setminus R} \phi(r) = 0$. Since $\phi(r) > 0$ for all r, it follows that $S \setminus R$ is an empty set. Thus $S = R$. Since S is a set of elements dividing $q-1$, $q-1 \in S$. It follows that $q-1$ is also in R, the set of orders of elements in F_q^*. Therefore, there is an element of order $q-1$ in F_q^*. Thus there exists at least one generator in F_q^*. □

The proof of the case of the existence of the generator in the multiplicatively written group $(\mathbb{Z}/p\mathbb{Z})^*$ where p is a prime is quite similar.

2.2. **Discrete Logarithm Problem.** The mathematics of the method behind Diffie-Hellman is based on the *discrete logarithm problem*. As was discussed before, public-key cryptography rests on the idea of one-way functions. One example could be an exponential function in a large finite field. It would not be a good candidate for a one-way function over the reals since it is not easier to compute x^y rather than to get its inverse $log_x y$. Working over the finite field like F_p, one can compute x^y for large x rather quickly, for example using repeated-squaring method. However, if there is an element $z = x^y$, computing $y = log_x z$ is way more difficult in the finite field. Because of the use of the finite field, this problem is called "discrete".

Definition 2.5. The discrete logarithm problem on the multiplicatively written group, $(\mathbb{Z}/p\mathbb{Z})^*$ is defined in the following way. Let $x, z \in (\mathbb{Z}/p\mathbb{Z})^*$ where z belongs to the cyclic subgroup generated by x. Find an integer y such that:

$$x^y = z \bmod p \tag{2.6}$$

2.3. **Diffie-Hellman Key Exchange.** Let's suppose Alice and Bob have never met before. Diffie-Hellman key exchange allows them to jointly establish a shared secret key over an insecure channel. The algorithm can be implemented in the multiplicatively written group of any finite field. In this example, we will consider the most common implementation in a group of a prime field, $(\mathbb{Z}/p\mathbb{Z})^*$. It is important that $g \in (\mathbb{Z}/p\mathbb{Z})^*$ is a generator since we want to make sure the generated shared key at the end received from a power of g is any element of $(\mathbb{Z}/p\mathbb{Z})^*$. Let's outline the process step by step:

(1) Alice and Bob agree on a prime modulus p and a generator g, which are publicly known;

(2) Alice selects a private random number a such that $1 < a < p-1$ and calculates $l = g^a \bmod p$ sending the result publicly to Bob;
(3) Then Bob selects his private random number b such that $1 < b < p-1$ and calculates $m = g^b \bmod p$ sending the result publicly to Alice;
(4) Alice takes Bob's public result m and raises it to the power of her private number obtaining $m^a \bmod p$;
(5) Bob takes Alice's public result l and raises it to the power of his private number obtaining $l^b \bmod p$;
(6) We notice that $m^a \bmod p = (g^b)^a \bmod p = (g^a)^b \bmod p = l^b \bmod p = s$ is a shared key.

We note that only a and b are private knowledge, all the other values used during the exchange are publicly available. Therefore, the possible third party, Eve, will only have to work with g, l and m to obtain $g^{ab} \bmod p$. However, it takes extremely long time to compute having only this knowledge. We note that if p is extremely large, then even the fastest world computers will be unable to find a such that $g^a \equiv l \bmod p$ given only l, p and g, a fact that restates the difficulty of discrete logarithm problem. Finally, it is important to mention that the use of the generator g in $(\mathbb{Z}/p\mathbb{Z})^*$ complicates the problem for Eve because the powers of g can be any element of the field, increasing the amount of possible choices for a key. Since the message can be arbitrarily large which may cause the slow asymmetric encryption process, the obtained secret shared key is used in the symmetric encryption which allows Bob and Alice to send messages across the same open communications channel. Then the regular version of ElGamal scheme is used to encrypt the symmetric key which is rather small compared to the message.

2.4. **Other Public Cryptosystems.** RSA is another public-key cryptosystem introduced just after Diffie-Hellman key exchange. The security of RSA is guaranteed by two mathematical problems: an integer factorization problem and an RSA problem which is conjectured to be equally difficult.

Definition 2.7. The integer factorization problem can be stated as follows. Given a number $N = pq$, where p and q are large primes, find p and q.

Definition 2.8. The RSA problem is stated as follows. Find m such that $c \equiv m^e (\bmod\ n)$ where (n, e) is an RSA public key and c is an RSA ciphertext.

We can see that both cases use the concept of trapdoor function where the computation is easy to process in one direction but difficult to reverse.

Unlike Diffie-Hellman, RSA involves the use of two different keys (public and private) at each side where one is used for encryption and the other one - for decryption. The main improvement that RSA has over Diffie-Hellman key exchange is that it provides signatures generated using hash functions which not only verify that the message was actually from a specific sender but also help to make sure that its contents were not tampered with by a third party. This property of RSA is especially important since anyone can use the recipient's public key to send him encrypted messages. The idea behind this feature is that it is computationally impossible to find the same output for multiple inputs in the hash function, and the third party would not be able to change the content of the message without having the hash value changed. Among the disadvantages of RSA is that this cryptosystem may not be a sustainable solution for low-powered devices such as mobile

phones on which a lot of cryptography is done these days. Multiplying two prime numbers can take way more time than expected on such devices, and as a result, the trapdoor function may get less reliable (the gap between the difficulties of doing computations in both directions shrinks) in the long term. Security can be maintained by increasing the key sizes, but in that case efficiency is largely compromised. Thus, there arises a need in a public-key system based on the convenient trapdoor function, the one providing a better balance between security and efficiency.

3. Elliptic Curve Cryptography

Researchers spent quite a lot of time trying to explore cryptographic systems based on more reliable trapdoor functions and in 1985 succeeded by discovering a new method, namely the one based on elliptic curves which were proposed to be the basis of the group for the discrete logarithm problem. Researchers believe that elliptic curves guarantee more security and provide with much smaller key sizes than other groups. For better understanding of the extent, let us use the visualization that compares the amount of energy one needs to break a cryptographic system with how much water that energy could boil. For example, a 228 - bit RSA key can be broken using less energy than that required to boil a teaspoon of water. However, one can equate the amount of energy needed to break a 228 - bit elliptic curve key with the energy used to boil all water on earth. A much longer RSA key of around 2,380 bits is needed for the same level of security which is rather inefficient. Let us dive deeper into what constitutes the mathematics behind this concept of elliptic curves.

3.1. **Elliptic Curve Fundamentals.**

Definition 3.1. An elliptic curve E over a field K is a cubic curve that consists of the points (x, y) satisfying the equation

$$y^2 = x^3 + ax + b \tag{3.2}$$

together with an element $\mathcal{O}$ called "the point at infinity".

It is important to note that (3.2) is in the simplified Weierstrass form and holds only for fields in which the characteristic is not equal to 2 or 3. The requirements that the discriminant $\Delta = -16(4a^3+27b^2)$ is nonzero and the polynomial x^3+ax+b has distinct roots ensure the curve's non-singularity.

3.2. **Elliptic Curves over the Reals.**

First, to get the general idea of how operations over elliptic curves work, we define the properties of elliptic curves over real numbers. The field of real numbers is used to get a clearer idea of the visual representations of the curves and understand how the geometry of the points on the curves works. An elliptic curve over the reals is defined by (3.2) where a and b are real numbers. The graph of the elliptic curve over real numbers consists of two components if its discriminant is positive and of one component if it is negative. We now define the group law on elliptic curves which is useful for cryptographic purposes. In this paper, we will use the geometric approach to introduce the group law. Let's suppose that $\mathcal{O}$ is a "point at infinity" and that all the vertical lines in the space where our elliptic curve exists go through this point. Let E be an elliptic curve and let P and Q be two points on E. The addition of the points on the curve is best illustrated by the following composition law. We will use $\oplus$ to denote the composition of two points.

Definition 3.3. If $P, Q \in E$ and L is the line through Q and P (if $P = Q$, L is the line tangent to E at P) which intersects the curve at the third point R, then the line L' through $\mathcal{O}$ and R intersects E at the third point which we denote $P \oplus Q$. Thus $P \oplus Q$ is the point we get as a result of adding points P and Q on the curve.

The above law can be visualized on the elliptic curve in $\mathbb{R}$ (see Fig. 1, 2).

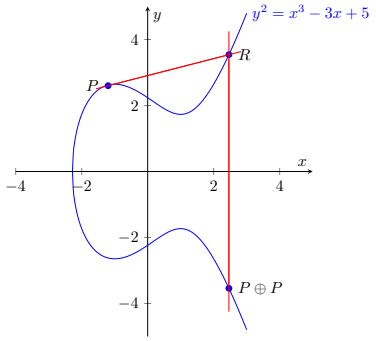


FIGURE 1. Adding a point to itself.

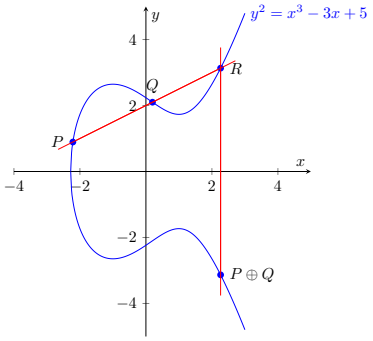


FIGURE 2. Adding two distinct points.

We now claim that the points on the elliptic curve form an abelian group. This can be derived from the properties of the composition law.

Proposition 3.4. *Let P, Q, R be three points (not necessarily distinct) lying on the intersection of E and a line L. Then the composition law has the following properties:*

- $(P \oplus Q) \oplus R = \mathcal{O}$.
- $P \oplus \mathcal{O} = P$ *for all* $P \in E$.
- *If* $P \in E$, *then there is a point* $\ominus P \in E$ *such that* $P \oplus (\ominus P) = \mathcal{O}$.
- $(P \oplus Q) \oplus R = P \oplus (Q \oplus R)$.

The above properties together with a closure property make E into abelian group.

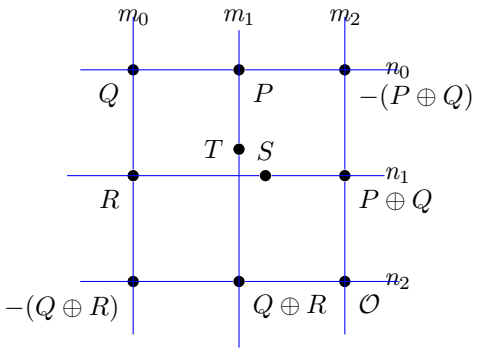
Proof. (1) By the composition law, $P \oplus Q$ is the point we get as a result of adding points P and Q on E. Let L' be the line through $\mathcal{O}, R$ and $P \oplus R$. By Bézout's Theorem, any line intersects the elliptic curve at three points counting multiplicity. Then the point we get as a result of adding $P \oplus Q$ and R is the third point of intersection of the line tangent to E at $\mathcal{O}$ with E. Let S be that point. Then for any collinear choice of P, Q and R, $(P \oplus Q) \oplus R = S$. In particular, we have that $(\mathcal{O} \oplus \mathcal{O}) \oplus \mathcal{O} = S$. By working through the construction of addition, $\mathcal{O} \oplus \mathcal{O} = \mathcal{O}$. Therefore, $S = \mathcal{O}$. This means that the third point of intersection of the line tangent to E at $\mathcal{O}$ is $\mathcal{O}$, and thus, $(P \oplus Q) \oplus R = \mathcal{O}$.

(2) If we let $Q = \mathcal{O}$ in the composition law and since every line is defined by two points, it can be seen that $L = L'$. Since L goes through the points $P, \mathcal{O}$ and R and L' - through $\mathcal{O}, R$ and $P \oplus \mathcal{O}$, $P \oplus \mathcal{O} = P$. Then $\mathcal{O}$ is the additive identity of E.

(3) Since P and Q both lie on L, we will end up at the same point irrespective of whether we add P and Q or Q and P. Thus, $P \oplus Q = Q \oplus P$.

(4) Let the line L'' intersect E at $P, \mathcal{O}$ and R. Then, by properties (1) and (2), $(P \oplus \mathcal{O}) \oplus R = \mathcal{O} = P \oplus R$. This proves that there exists a point $Q = \ominus P$ which is the additive inverse of $P \in E$.

(5) The proof of the associativity of the points on E applies only to the generic case but, nevertheless, gives a good idea of why the property holds. First, we assume that P, Q and R are on the elliptic curve E over a field K with a distinguished point $\mathcal{O}$. Moreover, we assume that P, Q and R are in "general position" which means that they have no special relations. In this case, this means that all the 8 points we construct below are distinct. By Bézout's Theorem and the property (1) of the composition law, the line m_0 through Q and R intersects E at the third point $-(Q \oplus R)$ and the line n_0 through P and Q intersects E at $-(P \oplus Q)$. Following the same argument, the line m_2 through $\mathcal{O}$ and $-(P \oplus Q)$ intersects E at $P \oplus Q$ and the line n_2 through $\mathcal{O}$ and $-(Q \oplus R)$ intersects E at $Q \oplus R$. Let S be the third point of intersection of the line n_1 through R and $P \oplus Q$ with E and let T be the third point of intersection of the line m_1 through P and $Q \oplus R$ with E. As a result of the operations above, we get the diagram that looks like the one below.



We assume that $S \neq P \oplus Q, S \neq R, T \neq Q \oplus R, T \neq P$. Following our previous construction method, $S = -((P \oplus Q) \oplus R)$ and $T = -(P \oplus (Q \oplus R))$. To prove that $(P \oplus Q) \oplus R = P \oplus (Q \oplus R)$, we can just show that $S = T$. We will suppose that $S \neq T$ and derive a contradiction. Let V be the set of all homogeneous polynomials in x, y and z of degree 3. Then V is a vector space over K. V has dimension 10 since there are 10 monomials in x, y, z of degree 3 which form a basis of V. Each of the six lines we constructed above can be represented by homogeneous polynomial of degree 1 in a projective space. Let $g(x, y, z) = n_0 n_1 n_2$ and $h(x, y, z) = m_0 m_1 m_2$. Since $m_0 m_1 m_2$ and $n_0 n_1 n_2$ are homogeneous polynomials of degree 1, g and h are homogeneous polynomials of degree 3. So $g, h \in V$. We will construct a new basis of V containing g and h. For convenience, we will rename the eight points $P, Q, R, P \oplus Q, -(P \oplus Q), Q \oplus R, -(Q \oplus R), \mathcal{O}$ as $X_0, ..., X_7$. For $i \geq 0$, let's construct a homogeneous polynomial ϕ_i of degree 3 such that $\phi_i(X_j) = 0$ if $i \neq j$ and $\phi_i(X_i) \neq 0$. This polynomial can be obtained by multiplying any three of the lines $m_0, ..., m_2, n_0, ..., n_2$ such that X_i does not lie on any of the three chosen lines and X_j lies on one of them. For $0 \leq i < 8$, let

$$\mathcal{B}_i = \{g, h\} \cup \{\phi_j(X_j) : 0 \leq j < i\}. \tag{3.5}$$

We claim that $\mathcal{B}_8$ is a basis of V. First, we will show that $\{g, h\}$ is linearly independent. We observe that $g(S) = n_0(S)n_1(S)n_2(S) = 0$. Since we claimed that $S \neq T$ and by Bézout's Theorem, S cannot lie on any of the lines $m_1, ..., m2$. So $h(S) = m_0(S)m_1(S)m_2(S) \neq 0$. Similarly, $h(T) = 0$ and $g(T) \neq 0$. Therefore, g and h are not multiples of each other. We observe that for every i such that $\varphi \in \mathcal{B}(i), \varphi(X_i) \neq 0$. Then $\phi_i \notin$ span $\mathcal{B}_i$ because $\phi_i(X_i) \neq 0$. Therefore, dim span $\mathcal{B}_{i+1} =$ dim span $\mathcal{B}_i + 1$. From our construction of the basis in (3.5), dim span $\mathcal{B}_0 = 2$. Hence dim span $\mathcal{B}_8 =$ dim span $\mathcal{B}_0 + 8 = 10$. Thus $\mathcal{B}$ is a basis of V. Suppose our elliptic curve E is represented by the Weierstrass equation of the form $F(x, y) = 0$. The curve is represented in the projective space by a smooth homogeneous polynomial $f(x : y : z) = z^3F(x/z, y/z)$ of degree 3. In particular, $f \in V$. Thus, we can write f as a linear combination of the elements of the basis $\mathcal{B}_8$:

$$f = ag + bh + \sum_{i=0}^{7} c_i\phi_i.$$

By construction, we have that $f(X_i) = c_i\phi_i(X_i)$ for each i. However, $f(X_i) = 0$ for all i since $X_0, ..., X_7$ lie on E. Since $\phi_i(X_i) \neq 0$, it follows that $c_i = 0$ for all i and so $f = ag + bh$. Similarly, we see that

$$0 = f(S) = ag(S) + bh(S)$$
$$0 = f(T) = ag(T) + bh(T).$$

Since $g(S) = h(T) = 0$ and $h(S) \neq 0, g(T) \neq 0$, it follows that $a = b = 0$. Hence, we conclude that $f = 0$ which is a contradiction since we defined f to be a polynomial of degree 3. Thus, our supposition was wrong and $S = T$ which means $(P \oplus Q) \oplus R = P \oplus (Q \oplus R)$.

□

In addition to the properties above, the points on the elliptic curve satisfy the closure property or in other words, if $P, Q \in E$, then $P \oplus Q \in E$. This can be verified by the composition law we defined before. Thus, we conclude that the points on the elliptic curve form an abelian group.

3.3. **Elliptic Curves over Finite Fields.** Let $K = F_q$ be the finite field where $q = p^r$ and E be an elliptic curve defined over K. An elliptic curve over F_q is defined by (3.2) where $a, b \in F_q$. The graph of elliptic curve over a finite field does not look as neat as the one over $\mathbb{R}$: usually, it is just a discrete set of points. Nevertheless, the set of points $E(F_q)$ still forms a finite abelian group in F_q which is either cyclic or a product of two cyclic groups. A group forms a finite number of points, a feature that enables precise arithmetic to be performed with faster calculations and lower probability of a round-off error. For example, the field F_p is widely used in practice mostly because of the convenience of mathematical operations used in it. All the values produced by operations on the points within a field are reduced modulo p and result in another points within the same field. Besides, every integer has a multiplicative inverse modulo p which is convenient for division in the field. An important information regarding elliptic curves over finite fields is the number of rational points it forms. The value of the number of points is essential for determining the difficulty of solving the discrete logarithm problem in $E(F_q)$ and ensures the security of the system which depends on that value having a large prime

factor. Since there are q choices for each x and for each choice of x, there are at most 2 choices for y in the equation that defines elliptic curves over the finite fields, E has at most $2q+1$ F_q points counting $\mathcal{O}$, "the point at infinity". Hasse's theorem gives a more precise estimate of the number of points on E bounding the value both above and below:

$$|\#E(F_q) - (q+1)| \leq 2\sqrt{q}.$$

It follows from the result that $\#E(F_q)$ grows approximately as q, the number of elements in the field. To get the exact number of points on the certain curve, Schoof's algorithm is widely used. It was the first polynomial-time algorithm in $\log q$ the running time of which was $O(\log^8 q)$. The approach makes use of the Hasse's theorem, Frobenius endomorphism ϕ, Chinese remainder theorem and division polynomials. For more details on this, see [7].

3.4. **Computing Large Multiples of a Point.** Let G denote the abelian group formed by the points on the elliptic curve E with a $\oplus$ operation we defined before. The multiplication by scalar t on E is defined by repeatedly adding the point P the amount of times that is the same as the value of the scalar:

$$tP = \underbrace{P \oplus P \oplus ... \oplus P}_{\text{t}}$$

The multiple of P can be computed more efficiently rather than by just adding P to itself t times which takes linear time. This can be done by the double and add method. First of all, to compute tP, we start with the binary expansion of t:

$$t = t_0 + 2t_1 + 2^2t_2 + ... + 2^kt_k,$$

where $t_0...t_k \in \{0, 1\}$. Then tP is computed as:

$$t = t_0P + 2t_1P + 2^2t_2P + ... + 2^kt_kP,$$

where we can compute 2^kP by k doublings and $k = \lfloor \log_2(t) \rfloor$. Assuming that on average, half of the terms in the expansion will be 0, tP can be computed with $\log_2(t)$ doublings and $\frac{1}{2}\log_2(t)$ additions. Algorithm 1 extends the idea presented above.

Let i represent the bit of t and m - the number of digits in the binary representation of t.

Algorithm 1 Double and Add

```
1: procedure DOUBLEANDADD(t, P)                ▷ The product of t and P
2:     result ← 0
3:     m ← math.floor(log_2(t)) + 1
4:     for i = 0 to m do                       ▷ Iterating through the binary digits of t
5:         if t_i = 1 then
6:             result ← result + P
7:         P ← 2 * P
8:     return result
```

Since there are a total of m steps in the algorithm and at most 2 operations per step, the worst case scenario is that we would have to perform $2m$ operations. This algorithm speeds up the computation time to $O(m)$ or equivalently $O(\log(t))$ which is exponentially better than the linear algorithm mentioned before.

3.5. **Elliptic Curve Discrete Logarithm Problem.**

Definition 3.6. Let E be an elliptic curve defined over a finite field and G - defined as before. If $P \in G$ and $R = kP$ is a multiple of P where k is a scalar, we define the discrete logarithm problem on E as follows. Find an integer k such that $kP = R$ where P is a generator point on E and R belongs to the cyclic subgroup generated by P .

The double and add algorithm we described before proved that computing kP can be rather fast and efficient. However, discrete logarithm problem on E is said to be much harder to solve than discrete logarithm problem in the multiplicatively written group $(\mathbb{Z}/p\mathbb{Z})^*$. The best known algorithms that can break discrete logarithm problem on elliptic curves have purely exponential runtime. This again supports the reliability of the trapdoor function that is the basis of elliptic curve cryptosystem.

3.6. **Elliptic Curve Diffie-Hellman (ECDH).** The idea behind the exchange system is the same as the one behind the regular Diffie-Hellman: Alice and Bob want to communicate securely over an insecure channel without a need to meet each other. Let's consider the most common implementation in the prime field, F_p.

(1) Alice and Bob agree on a set of domain parameters such as (p, a, b, P, n, h) where p is a prime, a and b are random values that make up the equation of elliptic curve E, P is a random point on E, n is the order of P and h is a cofactor of the group G. Order n is usually a prime number and is the smallest possible integer such that $nP = \mathcal{O}$. Then the cyclic subgroup of G generated by P is of the form:

$$\langle P \rangle = \{\mathcal{O}, P, 2P, 3P, ..., (n-1)P\}.$$

Since n is the order of P, it is the size of the subgroup generated by P. Thus it follows by Lagrange's Theorem that $h \cdot n = |G|$ where h is a cofactor and in this case, the number of disjoint cyclic subgroups formed by G. To make the discrete logarithm problem harder to solve, n should be a large number. On the other hand, h must be small ($h \leq 4$), preferably $h = 1$. The derivation of domain parameters is not usually done by each participant because it is rather time-consuming. The reason for this is that the process may require the computation of $|G|$ which is usually done by the involved Schoof's algorithm. That is why, several organizations publish recommended curves with computed parameters for participants to use. All the domain parameters are public knowledge.

(2) Both Bob and Alice randomly choose a private key d from the interval $[1, n-1]$ and calculate the public key $Q = dP$. The point P is used as a generator. Then Alice's key pair is (d_A, Q_A) and Bob's key pair is (d_B, Q_B).

(3) The parties exchange each other's public keys. Alice computes $d_A Q_B$ using Bob's public key and Bob computes $d_B Q_A$ using Alice's public key. The result calculated by both parties is equal since $d_A Q_B = d_A d_B P = d_B d_A P = d_B Q_A$ and is thus a shared key. The shared secret is the x or y-coordinate of the computed point $d_A d_B P$. A third party Eve only has knowledge of P, Q_A and Q_B and will be unable to get the shared key without solving the discrete logarithm problem.

The obtained shared key can be used to encrypt the communication between parties using symmetric-key cipher. Alternatively, a fully asymmetric cryptosystem based on the elliptic curve analogue of the basic ElGamal encryption scheme can be used for message transmission, even though it is usually slower than the symmetric one. Therefore, just like in regular Diffie-Hellman key exchange, it is used more just for key encryption. We present it in the following section.

3.7. **ElGamal System on Elliptic Curves.** Let p be a prime, E - a chosen elliptic curve over F_p, P - a randomly chosen point on E, and n - the order of P. Let Q be the public key of the intended recipient of the enciphered message generated in the same way as in Diffie-Hellman key exchange. Let m represent the plaintext.

First of all, a sender Alice comes up with a public fucntion $f : m \mapsto M$ which maps a message m to a point M on E. Then, she chooses a random value k such that $k \in_R [1, n-1]$ and computes $C = kP$. After that, she gets a point M on the curve by computing $M = f(m)$. Finally, she computes $D = M + kQ$. The ciphertext she sends to Bob is represented as a set of points (C, D).

Then Bob uses his private key d to get a plaintext. He first computes $M = D - dC$ and then performs $m = f^{-1}(M)$.

We note that $dC = d(kP) = k(dP) = kQ$ and it can be seen that a third party who wishes to receive a value of M, needs to compute kQ. Computing kQ given domain parameters kP and Q is the same discrete logarithm problem introduced as part of the Diffie-Hellman key exchange.

3.8. **Elliptic Curve Digital Signature Algorithm.** Another advantage of elliptic curve cryptography is that just like RSA cryptosystem, it provides the opportunity for the parties to "sign" their messages so that the receiver knows exactly the message is from him. Let m be the message and n be the prime order of the subgroup generated by P.

Algorithm 2 Signature generation algorithm

1: **procedure** SigGen(m, n, P)
2: Compute $e = hash(m)$.
3: $z \leftarrow l$ leftmost bits of e where l is bit length of n.
4: Select $k \in_R [1, n-1]$.
5: Compute $(x_1, y_1) = kP$.
6: Compute $r = x_1 \bmod n$.
7: **if** $r = 0$ **then**
8: Select a new k, back to step 4.
9: Compute $s = k^{-1}(z + rd_A) \bmod n$ where d_A is Alice's private key and k^{-1} is the multiplicative inverse of $k \bmod n$.
10: **if** $s = 0$ **then**
11: Select a new k, back to step 4.
12: Return (r, s). ▷ the signature

Then Bob can verify Alice's signature using Algorithm 3. He needs to obtain a copy of Alice's public key Q_A to accomplish this task.

Even though ECC shows many great advantages mentioned before, one if its main drawbacks is that the domain parameters need to be computed in advance

Algorithm 3 Signature verification algorithm

procedure SigVer(E, n, Q_A, s, z, P, r)
  if $Q_A \neq \mathcal{O}$ and $Q_A \in E$ and $nQ_A = \mathcal{O}$ and $r, s \in_R [1, n-1]$ **then**
    Compute $u_1 = zs^{-1} \bmod n$.
    Compute $u_2 = rs^{-1} \bmod n$.
    Compute the point $(x_1, y_1) = u_1P + u_2Q_A$.
    if $(x_1, y_1) \neq \mathcal{O}$ and $r \equiv x_1 \bmod n$ **then**
      "The signature is valid".
    else
      "The signature is invalid".
  else
    "The signature is invalid".

and they are rather expensive to generate. Thus a thorough analysis of the curves must be done. In addition, just like any other cryptosystem, ECC may face danger by classical and quantum attacks some of which will be discussed in the next section.

3.9. **Attacks on ECC and Pollard's rho algorithm.** Elliptic curve cryptography is subject to both classical and quantum attacks. Classical attacks are usually slow and require exponential running time to solve the elliptic curve discrete logarithm problem. Among quantum attacks, there are cases like Shor's algorithm which fulfills the attack in polynomial time. In this section, we will present the Pollard's rho-algorithm which is the fastest classical algorithm for solving ECDLP. Let E be the elliptic curve and the discrete logarithm problem on E be as defined in (3.6). Let n be the order of the subgroup generated by P. The running time of the algorithm is roughly $O(\sqrt{n})$. In addition, it requires just $O(1)$ in space complexity which is the best result compared to other methods that solve the discrete logarithm problem. The idea behind the algorithm is to find distinct pairs of integers (a_{j_1}, b_{j_1}) and (a_{j_2}, b_{j_2}) such that $a_{j_1}P + b_{j_1}Q = a_{j_2}P + b_{j_2}Q$. This can be done by the most efficient algorithm for this purpose: Floyd's cycle finding algorithm. One needs to partition the set of points on E into three subsets of roughly the same size and apply a suitable iterating function f to them. The result of applying the function to each point will generate the sequence with terms of the form $A_i = a_jP + b_jQ$. Once there is a match $A_{i_1} = A_{i_2}$, we get $a_{j_1}P + b_{j_1}Q = a_{j_2}P + b_{j_2}Q$. The match will be found eventually since the number of points on the curve is finite and the subgroup generated by P is cyclic. The birthday paradox ensures the high probability of that event. The path for finding $a_{j_1}, b_{j_1}, a_{j_2}, b_{j_2}$ will consist of a loop with a tail attached to it which looks just like letter ρ. In the end, we compute $k = (a_{j_1} - a_{j_2})(b_{j_2} - b_{j_1})^{-1} \bmod n$. According to research, the randomness of the function f and thus the performance of the algorithm can be improved by increasing the number of partitions of the set of points on E.

What concerns quantum attacks, Peter Shor in 1994 showed that because of their interesting property of computing over qubits, quantum computers can take polynomial time to factor large numbers or solve a discrete logarithm problem over a finite field. Scientists believe, that we will be able to use ECC more or less securely until quantum computers take over.

3.10. **Future of ECC.** We have mentioned that Shor's algorithm could destroy the elliptic curve cryptography as it is. Even though quantum computers still exist mostly in theory, a lot of organizations are thinking of ways to prevent unexpected attacks by moving towards inventing quantum-resistant schemes. Even though the life of the traditional elliptic curve cryptography may come to an end, recently Diffie-Hellman key exchange based on isogenies of super singular elliptic curves was developed which may prove to be quantum-resistant. This may take elliptic curve cryptography to a new, post-quantum level. We will not concern ourselves here with the details of implementation but will make some definitions and notes:

(1) The algorithm uses super singular curves over F_{p^2} where p is a prime;
(2) A supersingular curve is defined as having no points of order p;
(3) An isogeny $\phi : E_1 \to E_2$ is a rational map such that the number of points on the two curves is the same.

The reason for the belief in success of the algorithm is that the set of isogenies forms a non-abelian group. That is why, it is resistant to Shor's attack which targets algorithms based on abelian groups. The isogeny-based exchange provides small keys which is rather efficient but lacks further research which would fully prove its security.

Among classical algorithms, there also exist some which are based on non-abelian groups and thus are conjectured to be resistant to quantum attacks. Some of those are: hash-based cryptography, lattice-based cryptography, multivariate equations, and error codes.

Acknowledgments

"I would like to thank Peter May for organizing the 2020 Apprentice Program despite the difficult circumstances. I would also like to thank all of the REU staff and faculty for keeping the program interesting throughout the whole time. In particular, I am especially thankful to my mentors Cindy Tan and Iris Li for meeting with me frequently throughout the process and helping me out with any questions I had."

4. Bibliography

References

[1] Canteaut, Anne. Chapter 1. Finite Fields. https://www.rocq.inria.fr/secret/Anne.Canteaut/MPRI/ff.pdf
[2] Hankerson, Darrel; Menezes, Alfred; Vanstone, Scott. Guide to Elliptic Curve Cryptography. New York: Springer-Verlag. 2004. pdf.
[3] Koblitz, Neal. A Course in number Theory and Cryptography. New York: Springer - Verlag. 1994. pdf.
[4] Silverman, Joseph H. An Introduction to the Theory of Elliptic Curves. https://www.math.brown.edu/~jhs/Presentations/WyomingEllipticCurve.pdf
[5] Silverman, Joseph H. The Arithmetic of Elliptic Curves. Springer Science+Business Media, LLC 2009, 1986. pdf.
[6] Sutherland, Andrew. 18.783 Elliptic Curves. Lecture 2: Elliptic curves as abelian groups. https://math.mit.edu/classes/18.783/2015/LectureNotes2.pdf
[7] Sutherland, Andrew. 18.783 Elliptic Curves. Lecture 9: Schoof' algorithm. https://math.mit.edu/classes/18.783/2015/LectureNotes9.pdf
[8] Wohlwend, Jeremy. Elliptic Curve Cryptography: Pre and Post Quantum. https://math.mit.edu/~apost/courses/18.204-2016/18.204_Jeremy_Wohlwend_final_paper.pdf