Securing Data Transfer with Elliptic Curves

To what extent can elliptic curves be used to establish a shared secret over an insecure channel?

Mathematics

Word count: 3989

Contents

Introduction 3
Group Theory 4
$\mathbb{Z}_p^\times$: The Multiplicative Group Over a Prime 4
The Discrete Log Problem 4
Finite Field Cryptography and Attacks 5
Diffie-Hellman Key Exchange 5
Index Calculus 6
Elliptic Curve Cryptography 8
Proof of Associativity 12
Group of elliptic curve points 14
Elliptic Curve Diffie-Hellman 14
Finding the Discrete Log with Pollard's ρ algorithm 15
Evaluation 16
Diffie-Hellman in TLS 1.3 16
Finite Field Diffie-Hellman 17
Elliptic Curve Diffie-Hellman 17
Performance of group operations 17
Comparison 18
Conclusion 18
Bibliography 18

Introduction

As Alice tries to talk to Bob through her laptop, an adversary named Eve tries to eavesdrop their communication and steal sensitive information. Alice and Bob decide to use Diffie-Hellman Key Exchange, which allows them to establish a password for future communication, one that Eve cannot obtain even if she can intercept everything transmitted.

Diffie-Hellman is an important element in the collection of cryptographic methods that secure Internet connections. 99.3% of the top 1 million websites prefer using Diffie-Hellman over others when it comes to establishing a shared secret [1].

Much of cryptography is developed on the need to communicate messages securely in that other people cannot know what messages are being sent. Password-based, or *symmetric* cryptography uses a secret key known between two parties to communicate messages safely [2].

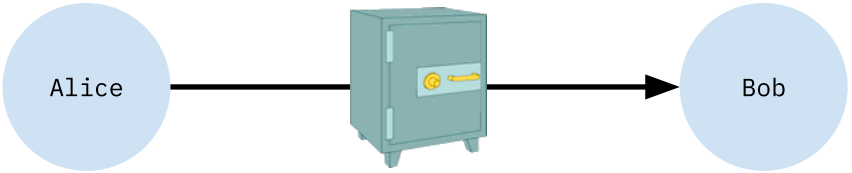


Figure 1: Diagram by author, derived from [3].

Under an analogy using safes, if Alice and Bob both know a secret combination number, they can send each other secret messages within safes configured with that combination, protecting their messages from being inspected by anyone else. Symmetric cryptography is then concerned with the mathematical processes that can be used to construct such a safe.

Diffie-Hellman arises from a need for Alice and Bob to quickly agree on a combination to use even if their own form of communication is one that can be eavesdropped by third parties like Eve [4]. This is at the core of securing Internet connections, since visitors of a website should not be forced to physically visit a company in order to establish a secret key for communication. Diffie-Hellman is designed specifically so that people can establish a shared secret (the combination between Alice and Bob) over an insecure channel (a communication method that Eve can eavesdrop).

Diffie-Hellman takes on different forms. There is Finite Field Diffie-Hellman and Elliptic Curve Diffie-Hellman. To answer our research question, we'll compare the two methods in terms of how efficient they are (how much data do Alice and Bob need to send to each other?) and how fast they are (how quickly can Alice and Bob calculate the shared password in an exchange?) to show the effectiveness of elliptic curves.

Group Theory

Much of the math required to understand elliptic curves and how they help establish shared secrets utilizes group theory. This section is a short discussion on it.

$\mathbb{Z}_p^\times$: The Multiplicative Group Over a Prime

Fermat's Little Theorem suggests the following to be true for any non-zero integer a and prime p:

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

Extracting a factor of a, we get:

$$a \cdot a^{p-2} \equiv 1 \pmod{p}$$

Thus, under multiplication modulo p, any non-zero integer a multiplied by a^{p-2} results in 1. As 1 is the multiplicative identity ($1 \cdot x = x$), a^{p-2} is said to be a's *multiplicative inverse*. Consider the set of numbers from 1 to $p-1$. Every number a has a multiplicative inverse (a^{p-2}) modulo p; the set contains an identity element (1); multiplication is associative ($a \cdot (b \cdot c) = (a \cdot b) \cdot c$); and each multiplication will always result in a number between 1 to $p-1$ since it is performed modulo p (closure). These properties, existence of an identity element and inverses, associativity, and the closure of operations, are exactly the properties that define a group [5, pp. 73-76].

A group is a set equipped with an operation. Therefore, the $\in$ symbol and the word *element* applies to groups too. We will refer to the specific group we discussed above as $\mathbb{Z}_p^\times$. In a similar vein, $\mathbb{Z}_p^+$ specifies addition modulo p as its group operation.

The *order* of a group refers to the number of elements in that group. For $\mathbb{Z}_p^\times$, the order is $p-1$ since its elements are $1, 2, ..., p-1$. Hence $|\mathbb{Z}_p^\times| = p-1$. The additive group includes zero as the identity, therefore $|\mathbb{Z}_p^+| = p$.

The *order* of a specific element x, refers to the smallest integer k such that $x^k = 1$, where 1 is the identity element. For example, the order of 17 in $\mathbb{Z}_{1009}^\times$ is 1008, because $a = 1008$ is the smallest a such that $17^a \equiv 1 \pmod{1009}$, whereas the order of 2 in $\mathbb{Z}_{1009}^\times$ is 504, since $b = 504$ is the smallest b such that $2^b \equiv 1 \pmod{1009}$. Therefore, $|17| = 1008$ and $|2| = 504$.

As groups are used extensively in cryptographic techniques, the order of groups and elements becomes important as it is related to the difficulty of obtaining a shared secret secured using specific groups and elements.

The Discrete Log Problem

Under $\mathbb{Z}_{1009}^\times$, we are asked to find the smallest integer n for which $17^n \equiv 24$. In this case,

$$17^{456} \equiv 24 \pmod{1009}$$

And 456 is the first exponent for which the equivalence holds. Therefore $n = 456$ is the solution. More generally, the discrete log problem (DLP) asks for a smallest exponent n in a group G and $a, b \in G$ such that $a^n = b$.

The brute-force approach to this problem would be repeatedly performing the group multiplication, calculating a^2, a^3, a^4 and checking if any of them matches b. In the example problem, it would take 455 multiplications before finally arriving at the answer. If we tried to solve DLP repeatedly with the exponent n taken at random, brute-forcing would take on average $\frac{1}{2}|a|$ operations since answers are in the range of 0 to $|a|-1$. As the order $|a|$ gets big (towards numbers as big as 2^{200}), this approach quickly becomes infeasible.

Assuming that DLP is non-trivial to solve, exponentiation of elements in $\mathbb{Z}_n^\times$ can be characterized as a one-way function, where computing exponentiation is trivial, but finding the discrete log (the inverse) is much more difficult [6]. Building upon this property, we will show how it can be used to ensure information reaches the right person with cryptography.

Finite Field Cryptography and Attacks

Diffie-Hellman Key Exchange

Building off the previous example, suppose we're given the numbers 17, 407 and 24 in $\mathbb{Z}_{1009}^\times$. Assume

$$17^a \equiv 407 \pmod{1009}$$
$$17^b \equiv 24 \pmod{1009}$$

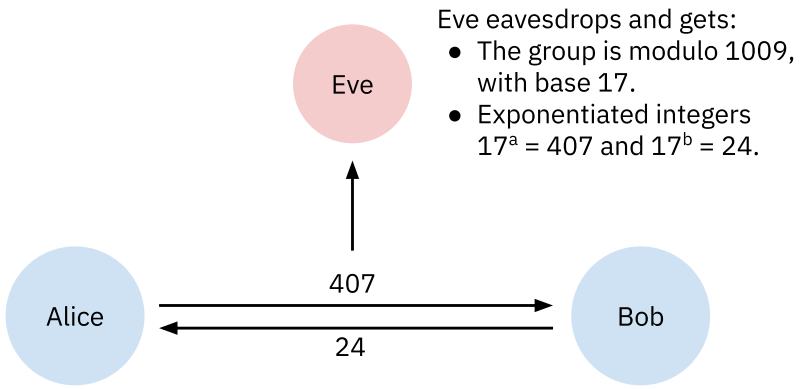
Can we find 17^{ab}? If we know $a = 123$, we can raise 24 to 123.

$$17^b \equiv 24 \pmod{1009}$$
$$a = 123$$
$$17^{ab} \equiv \left(17^b\right)^a \equiv 24^{123} \equiv 578 \pmod{1009}$$

More generally, if we know 17^a and the exponent b, or if we know 17^b and the exponent a, it would be possible for us to know 17^{ab}. However, given just 17^a and 17^b, there is no trivial way to find the answer.

Finding x^{ab} when just given x, x^a, and x^b is named the Diffie-Hellman problem. The difficulty of this problem relates to the difficulty of DLP, since solving the Discrete Log would give us the answer. Assuming the Diffie-Hellman problem is difficult, we can use this to setup a cryptographic exchange.

Consider the following case within $\mathbb{Z}_{1009}^\times$ where anything sent between Alice and Bob can be seen by Eve. Alice calculates $17^{123} \equiv 407$, and only sends Bob 407. Bob calculates $17^{456} \equiv 24$, and only sends Alice 24. After exchanging their information, Alice can compute $24^{123} \equiv 578$, and Bob can also compute $407^{456} \equiv 578$. Eve, only intercepting the numbers $17, 407, 24$ in their communication, is unable to calculate the secret number 578 without solving the Diffie-Hellman problem.



1. Alice generates 123, and computes $17^{123} = 407$. Alice sends 407 to Bob.
2. After receiving 24 from Bob, Alice computes $24^{123} = 578$.

1. Bob generates 456, and computes $17^{456} = 24$. Bob sends 24 to Alice.
2. After receiving 407 from Alice, Bob computes $407^{456} = 578$.

Figure 2: Diagram by author, a description of the Diffie-Hellman Key Exchange process.

To generalize, the Diffie-Hellman Key Exchange utilizes the difficulty of the Diffie-Hellman problem. Under an agreed upon group G and base $x \in G$, Alice and Bob can establish a shared secret. Alice can generate a secret exponent a and send Bob x^a, while Bob can generate a secret exponent b and send Alice x^b. Together, they can both compute x^{ab} as their shared secret securely, even if Eve is able to intercept this communication.

The specific example was done in $\mathbb{Z}_{1009}^{\times}$. In reality, the size of the group will be much larger to prevent anyone from trivially breaking the exchange and finding the secret [7], [8]. The more general technique of performing Diffie-Hellman on the group of multiplication modulo a prime is called Finite Field Diffie-Hellman. We first consider one way to break this, and afterwards show that elliptic curves would take more effort to break.

Index Calculus

Diffie-Hellman Key Exchange with modular multiplication normally uses groups $\mathbb{Z}_p^{\times}$ where $2^{2048} < p < 2^{8192}$ [7], [8]. The large size of the prime ensures that solving DLP is inefficient. For smaller groups, the Diffie-Hellman Key Exchange is less secure because DLP can be solved faster using Index Calculus.

Let's solve the following:

$$17^n \equiv 24 \pmod{1009}$$

To find n, we first define a logarithm function L. $L(x)$ is defined such that

$$17^{L(x)} \equiv x \pmod{1009}$$

Therefore, our goal is to find $L(24)$.

With

$$\begin{aligned} 17^{L(x)+L(y)} &\equiv 17^{L(x)} \cdot 17^{L(y)} \equiv xy \equiv 17^{L(xy)} \pmod{1009} \\ 17^{L(x)+L(y)} \cdot 17^{-L(xy)} &\equiv 17^{L(x)+L(y)-L(xy)} \equiv 1 \pmod{1009} \end{aligned}$$

Since $|17| = 1008$, we have (notice the change in modulus)

$$\begin{aligned} L(x) + L(y) - L(xy) &\equiv 0 \pmod{1008} \\ L(x) + L(y) &\equiv L(xy) \pmod{1008} \end{aligned}$$

As such, we have a relation analogous to the laws of logarithm on real numbers. Since every number can be factorized into primes, the idea is to obtain $L(p)$ for small primes p, then figure out $L(24)$ afterwards. We first try to factorize exponents of 17 into primes up to 13:

$$\begin{aligned} 17^{15} &\equiv 2^2 \cdot 5 \cdot 13 \pmod{1009} \\ 17^{16} &\equiv 2^7 \cdot 3 \pmod{1009} \\ 17^{24} &\equiv 2 \cdot 11^2 \pmod{1009} \\ 17^{25} &\equiv 2 \cdot 3 \cdot 13 \pmod{1009} \\ 17^{33} &\equiv 2^2 \cdot 3^2 \cdot 11 \pmod{1009} \\ 17^{36} &\equiv 2^2 \cdot 7^2 \pmod{1009} \end{aligned}$$

Applying L to both sides of the congruences, we obtain

$$\begin{aligned} 15 &\equiv 2L(2) + L(5) + L(13) \pmod{1008} \\ 16 &\equiv 2L(7) + L(3) \pmod{1008} \\ 24 &\equiv L(2) + 2L(11) \pmod{1008} \\ 25 &\equiv L(2) + L(3) + L(13) \pmod{1008} \\ 33 &\equiv 2L(2) + 2L(3) + L(11) \pmod{1008} \\ 36 &\equiv 2L(2) + 2L(7) \pmod{1008} \end{aligned}$$

There are six unknowns $L(2), L(3), L(5), L(7), L(11), L(13)$ and six congruences, using linear algebra methods, we can arrive at the solution

$$L(2) = 646, L(3) = 534, L(5) = 886, L(7) = 380, L(11) = 697, L(13) = 861$$

Next, the idea is to find $17^x \cdot 24$ for some x that can factorize over primes up to 13. (Note that $x = 0$ works here since 24 can be easily factorized, but in most cases it won't) Indeed, we have $17^2 \cdot 24 \equiv 2 \cdot 3^2 \cdot 7^2 \pmod{1009}$, so then we have

$$\begin{aligned} L(17^2 \cdot 24) &\equiv L(2 \cdot 3^2 \cdot 7^2) \pmod{1008} \\ 2 + L(24) &\equiv L(2) + 2L(3) + 2L(7) \pmod{1008} \\ L(24) &\equiv -2 + 646 + 2 \cdot 534 + 2 \cdot 380 \pmod{1008} \\ L(24) &= 456 \end{aligned}$$

We arrive at the answer $n = 456$.

Index Calculus finds n in $a^n \equiv b \pmod{p}$ [9, pp. 144-146] with this procedure:

- Find a^i for various i that can be factorized with small primes up to a certain limit (in our example, the factor base was up to 13).
- Solve $L(p_n)$ for these small primes with system of equations.
- Calculate $a^x \cdot b$ for various x until it can be factored with our factor base. Then find $L(b)$ by the following procedure:

$$a^x \cdot b \equiv p_1^{c_1} \cdot p_2^{c_2} \cdot ... \cdot p_n^{c_n} \pmod{p}$$
$$x + L(b) \equiv c_1 L(p_1) + c_2 L(p_2) + ... + c_n L(p_n) \pmod{|a|}$$
$$L(b) \equiv -x + c_1 L(p_1) + c_2 L(p_2) + ... + c_n L(p_n) \pmod{|a|}$$

This method relies on prime factorizations always existing for integers, which in general does not apply to ellpitic curve points [9, pp. 154-157].

For the example above, we examined exponents of 17 up to 17^{36}, and also computed $17 \cdot 24$ and $17^2 \cdot 24$. This took significantly less time than enumerating 17^n for all n until we reach 456. Therefore, Index Calculus is much more efficient than brute-forcing.

An improved method called General Number Field Sieve, based on Index Calculus, is more efficient than the latter for large primes [10]. The expected running time for the GNFS algorithm is proportional to [11]:

$$\exp\left((64/9)^{1/3}(\ln p)^{1/3}(\ln \ln p)^{2/3}\right)$$

where p is the prime that defines the group $\mathbb{Z}_p^\times$. Assuming that for $p = 3$ the GNFS runs in one nanosecond or 10^{-9} seconds, solving DLP for $p = 2^{2048}$ with GNFS would take about

$$\frac{\exp\left((64/9)^{1/3}(\ln 2^{2048})^{1/3}(\ln \ln 2^{2048})^{2/3}\right)}{\exp\left((64/9)^{1/3}(\ln 3)^{1/3}(\ln \ln 3)^{2/3}\right)} \cdot 10^{-9} \approx 1.02 \cdot 10^{26} \text{ seconds}$$

or

$$1.02 \cdot 10^{26} \cdot \frac{1}{60} \cdot \frac{1}{60} \cdot \frac{1}{24} \cdot \frac{1}{365.2425} \approx 3.22 \cdot 10^{18} \text{ years}$$

In later sections, we will take a look at Diffie-Hellman done on elliptic curves and how the time needed to solve DLP on elliptic curves compares with Diffie-Hellman on finite fields.

Elliptic Curve Cryptography

The equation $C : y^2 = x^3 + Ax + B$ (named the short Weierstrass form), where A and B are constants, defines an elliptic curve C. Elliptic curves are symmetric about the x-axis, since

$$(x, y) \in C \iff (x, -y) \in C$$

Elliptic curves can be over a field, which is a set that forms a group under addition, forms a group under multiplication with non-zero elements, and where multiplication is distributive:

$a(b+c) = ab + ac$ [12], [13]. Finite fields denoted as $\mathbb{F}_p$ can be represented with integers 0 to $p-1$ with addition and multiplication modulo p [14]. A curve is over a finite field $\mathbb{F}_p$ if $x, y \in \mathbb{F}_p$.

Since Diffie-Hellman works on any group, if we can show that points on an elliptic curve form a group, we can use elliptic curve and their points to establish shared secrets too. Furthermore, we will show that using ellptic curve points as the group for Diffie-Hellman provides quantifiable benefits.

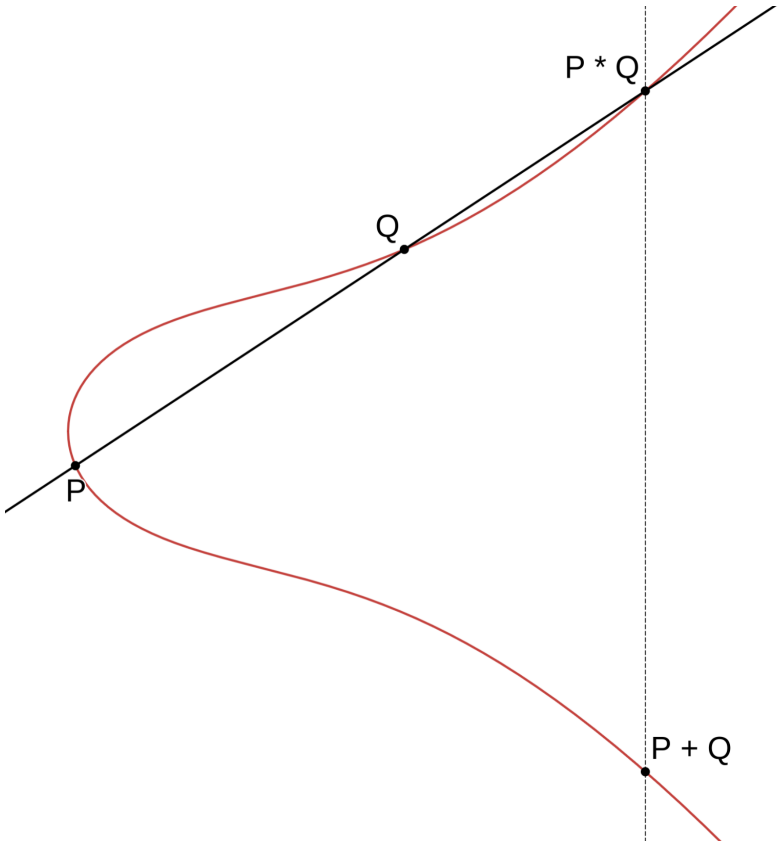


Figure 3: Diagram by author

Since a group requires a binary operation, starting with two points P and Q, one might try drawing a line between them and finding the third point of intersection on the curve. This can be denoted as $P * Q$. However, $*$ does not create a group. With $+$ computed as flipping the y-coordinate of the resulting point however, a group can be defined.

We now summarize the steps for defining a group law with $+$ as shown in [9, pp. 12-15].

Let $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ be distinct points on the curve, where $x_1 \neq x_2$. We can find a new point on the curve by drawing a line that goes across the two points, with

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

$$y = m(x - x_1) + y_1$$

We substitute this into the equation of the curve and try to solve for x:

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

Expanding and rearranging gives:

$$x^3 - m^2x^2 + (2m^2x_1 - 2y_1m + A)x + 2y_1mx_1 - m^2x_1^2 - y_1 = 0$$

With Vieta's formulas, the sum of roots for the cubic is m^2. We already know two distinct roots of this polynomial as x_1 and x_2, so we can find the x coordinate of the third point:

$$x_3 = m^2 - x_1 - x_2$$

To find the y-coordinate, we use the original line equation, but flip the resulting y-coordinate [15, p. 12].

$$y_3 = -(m(x_3 - x_1) + y_1) = m(x_1 - x_3) - y_1$$

Therefore, we have arrived at $R = (x_3, y_3)$, a third point distinct from P and Q.

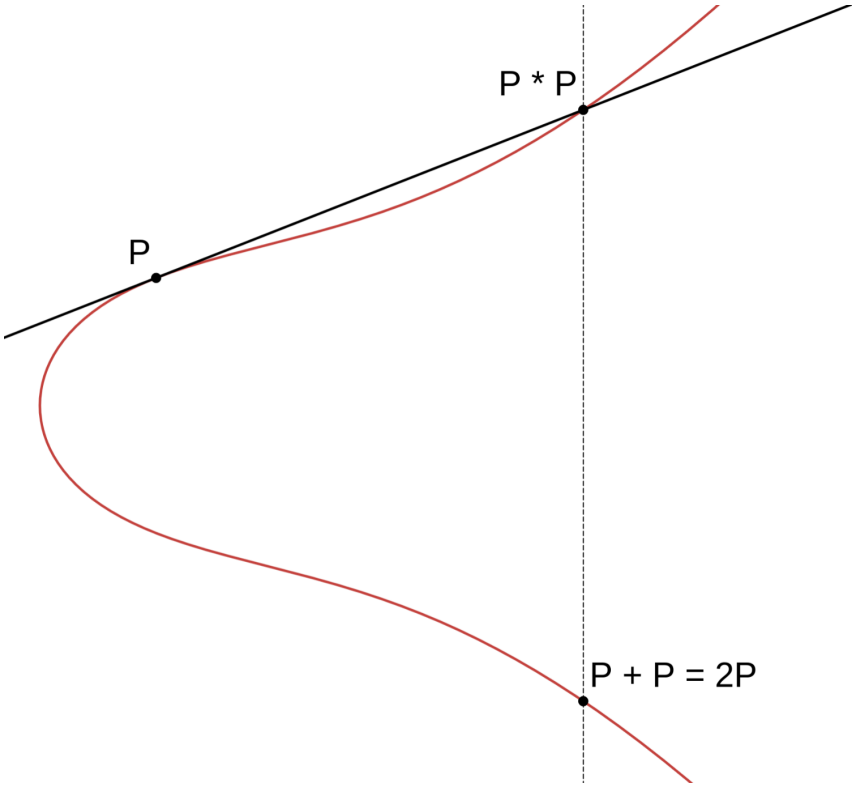


Figure 4: Diagram by author

If one point $P = (x_1, y_1)$ is known, we can use implicit differentiation to find the tangent line:

$$y^2 = x^3 + Ax + B$$

$$2y\frac{\mathrm{d}y}{\mathrm{d}x} = 3x^2 + A$$

$$m = \frac{\mathrm{d}y}{\mathrm{d}x} = \frac{3x^2 + A}{2y} = \frac{3x_1^2 + A}{2y_1}$$

With the same line equation $y = m(x - x_1) + y_1$, with the same expanded formula:

$$x^3 - m^2x^2 + (2m^2x_1 - 2y_1m + A)x + 2y_1mx_1 - m^2x_1^2 - y_1 = 0$$

This time, the line only intersects with two points on the curve, as it intersects with P at a tangent, so x_1 is a repeated root in the equation.

We can find the third point with

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

We can then develop a group law for points on elliptic curves. For special cases, such as adding two points on a vertical line, a "point at infinity" is added to the normal set of points on the curve, so that the group is well-defined for operations for all elements. This is studied more rigorously in projective geometry, though we incorporate this concept for simply defining the group law on elliptic curves [9, p. 11].

For an elliptic curve $C : y^2 = x^3 + Ax + B$ over a field $\mathbb{F}$ we define the following set:

$$E(C) = \{\mathbf{0}\} \cup \{(x, y) \mid y^2 = x^3 + Ax + B,\ \ x, y \in \mathbb{F}\}$$

Where $\mathbf{0}$ is the "point at infinity". We now show that $E(C)$ forms a group.

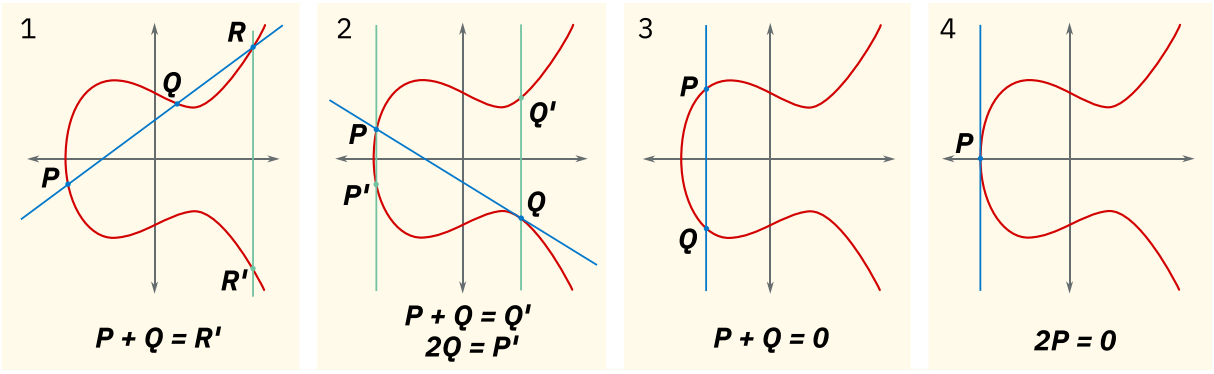


Figure 5: Diagram from [16], an illustration of the group operation defined on elliptic curves

Let P and Q be two points from $E(C)$. Define $+$ to be as follows:

- If $P = Q = (x_1, 0)$, let $P + Q = \mathbf{0}$. (#4 from Figure 5)
- If $P = Q = (x_1, y_1)$ where $y \neq 0$ (#2 from Figure 5), let

$$m = \frac{3x_1^2 + A}{2y_1}$$
$$x_3 = m^2 - 2x_1$$
$$P + Q = (x_3, m(x_1 - x_3) - y_1)$$

- If $P = (x_1, y_1), Q = (x_2, y_2)$ where $x_1 = x_2, y_1 \neq y_2$ (#3 from Figure 5): let $P + Q = \mathbf{0}$.

- If $P = (x_1, y_1), Q = (x_2, y_2)$ (#1 from Figure 5), let

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

$$x_3 = m^2 - x_1 - x_2$$

$$P + Q = (x_3, m(x_1 - x_3) - y_1)$$

- If $Q = \mathbf{0}$, $P + Q = P$.
- If $P = \mathbf{0}$, $P + Q = Q$.

Proof of Associativity

Perhaps the most surprising result of defining this operation is that the operation is associative, that is, $(P + Q) + R = P + (Q + R)$ for any three points $P, Q, R \in E(C)$. If we can prove that the $+$ operation is associative, then we can show that $E(C)$ forms a group, and Diffie-Hellman can be performed on $E(C)$. Proving this algebraicly becomes tedious, but there is a geometric argument from [15, pp. 8-15] using cubic space curves and Bézout's theorem for a specific case where the points have distinct x coordinates which is outlined below.

A cubic space curve is given with the following formula:

$$ax^3 + bx^2y + cxy^2 + dy^3 + ex^2 + fxy + gy^2 + hx + iy + j = 0$$

Therefore, an elliptic curve $x^3 + Ax + B - y^2 = 0$ is a cubic space curve. The union of three lines

$$(y - (m_1x + b_1))(y - (m_2x + b_2))(y - (m_3x + b_3)) = 0$$

is also a cubic space curve.

A consequence of Bézout's theorem is that two cubic space curves intersect at 9 points. The nine points could include the point at infinity in projective geometry, counting multiplicities as more than one point of intersection such as when at a tangent, and allows complex numbers as coordinates. For simplicity, the proof ignores these technicalities.

Proposition: Let C, C_1, C_2 be cubic space curves. Suppose C goes through eight of the nine intersection points between C_1 and C_2. Then C also goes through the ninth intersection point.

Proof: A total of 10 coefficients were used in the formula for a cubic space curve: $a, b, c, d, e, f, g, h, i, j$. Since scaling the equation by a linear factor results in the same curve, the curve can be said to be constrained by nine linear factors, or 9-dimensional. Suppose we constrained the curve such that it needs to go through a specific point. The number of linear factors that are allowed to vary would be decreased by one (similar to interpolating polynomials where adding another point requires a higher degree polynomial, this is in the other direction). If we constrain a cubic curve to go through eight specific points, the number of linear factors that are free to vary would be $9 - 8 = 1$. In other words, the set of all cubic curves that go through eight specific points is one-dimensional.

Let C_1 be specified by the equation $F_1(x, y) = 0$ and C_2 by $F_2(x, y) = 0$. We know that the set of all possible curves C that go through eight intersection points between C_1 and C_2 is one-dimensional.

A linear combination of F_1 and F_2 results in a cubic space curve that goes through the eight intersection points: $F_3 = \lambda_1 F_1 + \lambda_2 F_2$ with $F_3(a, b) = 0$ since any intersection point (a, b) will satisfy $F_1(a, b) = F_2(a, b) = 0$. F_3 is also free to vary by a single linear factor λ_2/λ_1 (accounting for scaling) so it represents a one-dimensional family of cubic space curves. Since both C and F_3 are one-dimensional, all curves C can be represented with the form $F_3(x, y) = 0$ for some λ_1 and λ_2.

The ninth intersection point also satisfies $F_3(x, y) = 0$, therefore C must go through the ninth intersection point. □

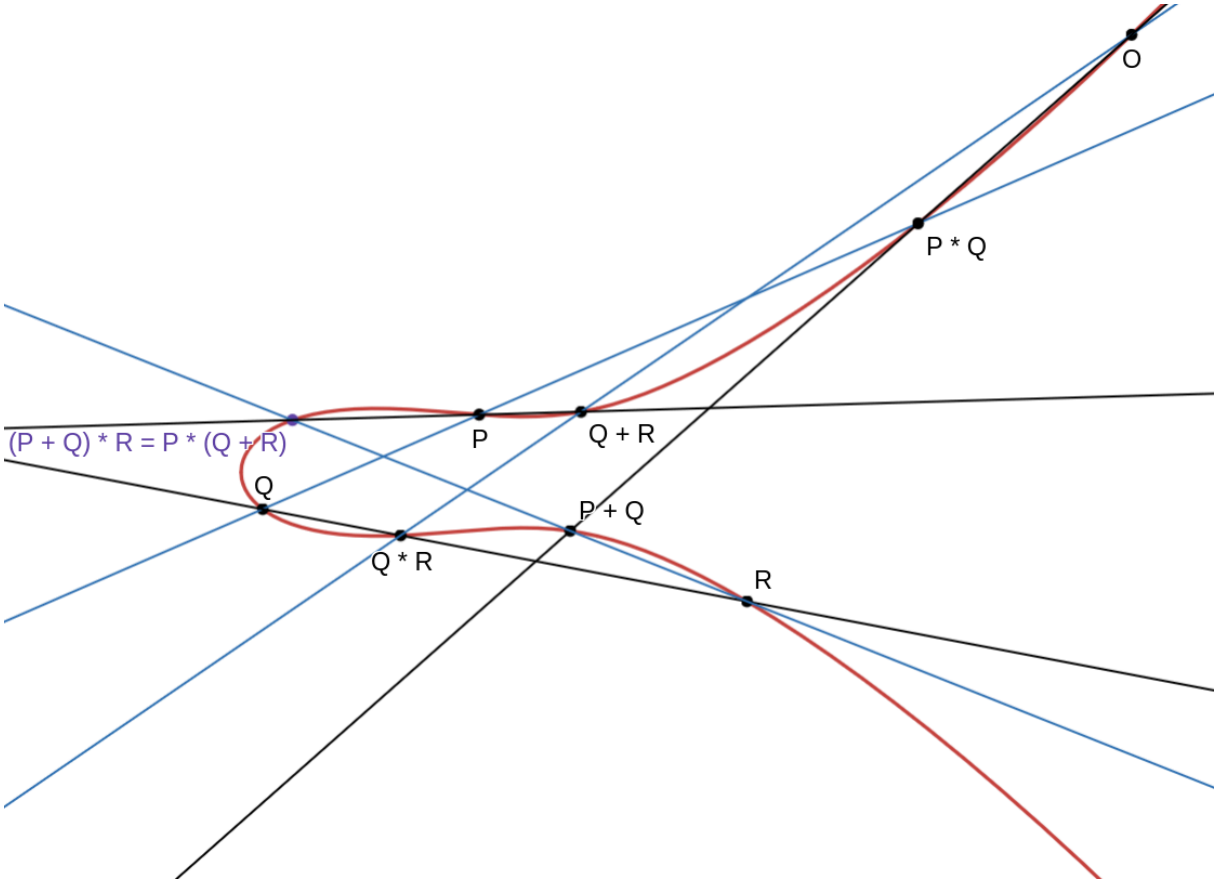


Figure 6: By author, graphical proof of associativity

Normally, the point at infinity is not shown in diagrams, but since it is treated as one of the points of intersection in Bézout's theorem, the point will be denoted as O.

Proposition: Let $P, Q, R \in E(C)$ with binary operations $+$ and $*$ as defined previously. Then $(P + Q) * R = P * (Q + R)$. (flipping the y-coordinate of points on both sides would give $(P + Q) + R = P + (Q + R)$, proving the associativity of $+$)

Proof: With Figure 6, we start with three arbitrary points on the elliptic curve, P, Q, and R. By drawing a black line, we find $Q * R$ from Q and R. $P * Q$ is found from P and Q with a blue line. To find $Q + R$, we draw a blue line from $Q * R$ to O, then take the third intersection point,

equivalent to “flipping” the intersection point used in the definition of elliptic curve addition. We find $P + Q$ by drawing a black line that goes through $P * Q$ and O.

A blue line is drawn from $P + Q$ and R to find $(P + Q) * R$. A black line is drawn from P and $Q + R$ to find $P * (Q + R)$.

We can group the union of three blue lines as C and group the union of three black lines as C_2, and let C_1 be the elliptic curve. Note that both C_2 and C intersect with C_1 at P, Q, R, $Q * R$, $Q + R$, $P * R$, $P + Q$, and O by the way we have drawn the lines. The only difference is that the ninth intersection between C_1 and C is known as $(P + Q) * R$, where the ninth intersection between C_1 and C_2 is known as $P * (Q + R)$.

Since C, C_1, C_2 are cubic space curves, we can use the proposition from earlier. C (the blue lines) goes through eight of the nine intersection points between C_1 (the ellptic curve) and C_2 (the black lines). C must also go through the ninth intersection point between C_1 and C_2 known as $P * (Q + R)$. From drawing the blue lines, the ninth intersection point is also known as $(P + Q) * R$, so it must be that $(P + Q) * R = P * (Q + R)$. □

This is deliberately an incomplete proof, since we did not cover corner cases such as when two of the three points have the same x-coordinate, a more complete proof can be seen in [9, pp. 20-32], we will assume associativity is true in general here.

Group of elliptic curve points

Therefore, $E(C)$ forms a group:

1. The operation $+$ is well-defined, closed and associative.
2. $\mathbf{0}$ is the identity, where $P + \mathbf{0} = P$ for all $P \in E(C)$.
3. Existence of inverse: let $P = (x, y)$, its inverse is $-P = (x, -y)$ since $(x, y) + (x, -y) = \mathbf{0}$.

Now that we have shown that elliptic curve points form a group, we can apply the Diffie-Hellman Key Exchange to elliptic curves as well.

Elliptic Curve Diffie-Hellman

Elliptic curve operations and modular multiplicative group operations differ in notation. In elliptic curves, the operation is commonly represented as addition of two points. Therefore $A + B$ is the normal operation on two points A and B while kA is the operation repeated ($2A = A + A$). In the multiplicative group modulo p, it corresponds to AB and A^k. Thus, an operation in previous sections such as A^k will now be written as kA in the context of elliptic curves.

With that note, Diffie-Hellman in elliptic curves follows the exact same procedure: two parties agree on a curve group to use, then decide on a base point P. Alice generates a secret integer a and sends Bob aP. Bob generates a secret integer b and sends Alice bP. They can now both calculate abP, which cannot be known by third parties unless they can solve the discrete log problem in elliptic curves.

A short example is as follows: Alice and Bob agree to use the curve $y^2 = x^3 + 6692x + 9667$ in $\mathbb{F}_{10037}$, with the base point $P = (3354, 7358)$ (from [15, p. 164]). Alice generates $a = 1277$ and sends $Q = aP = (5403, 5437)$ to Bob. Bob generates $b = 1337$ and sends $R = bP =$

$(7751, 1049)$ to Alice. Alice calculates $aR = abP = (8156, 1546)$, and Bob calculates $bQ = baP = (8156, 1546)$ as their shared secret.

To figure out this shared secret, Eve could try to break the discrete log for $Q = aP$.

Finding the Discrete Log with Pollard's ρ algorithm

Pollard's ρ algorithm is a general algorithm for solving the discrete log problem for any Abelian (commutative) group. It is less efficient than the general number field sieve on discrete log in finite fields, taking $O\left(\sqrt{N}\right)$ time on average in a group G where $|G| = N$ [9, pp. 147-150].

Pollard's ρ can be used to solve the problem above: for the curve $y^2 = x^3 + 6692x + 9667$ in $\mathbb{F}_{10037}$, with $P = (3354, 7358)$, $Q = (5403, 5437)$. Find k such that $kP = Q$.

Generate 10 random points on the curve based on multiples of P and Q:

$$M_0 = 42P + 37Q \qquad M_1 = 21P + 12Q \qquad M_2 = 25P + 20Q$$
$$M_3 = 39P + 15Q \qquad M_4 = 23P + 29Q \qquad M_5 = 45P + 25Q$$
$$M_6 = 14P + 37Q \qquad M_7 = 30P + 12Q \qquad M_8 = 45P + 49Q \qquad M_9 = 40P + 45Q$$

Then pick, in the same way, a random initial point:

$$A_0 = 15P + 36Q = (7895, 3157)$$

Then, choose an M_i point to add to based on the ones digit of the x coordinate of the point. As A_0 has $x = 7895$, $A_1 = A_0 + M_5 = (7895, 3157) + (5361, 3335) = (6201, 273)$.

Formally, define

$$A_{n+1} = A_n + M_i \text{ where } i \equiv x_n \pmod{10}$$

for $A_n = (x_n, y_n)$. The choice of random M_i points creates a kind of "random walk" of the points in the elliptic curve. As we keep calculating, we get:

$$A_0 = (7895, 3157), A_1 = (6201, 273), ...,$$
$$A_{95} = (170, 7172), A_{96} = (7004, 514), ...,$$
$$A_{100} = (170, 7172), A_{101} = (7004, 514)$$

We reach a cycle with $A_{95} = A_{100}$. Since we know the multiples of P and Q for all of the M_i points and thus all A_n points, keeping track of them gives us $A_{95} = 3126P + 2682Q$ and $A_{100} = 3298P + 2817Q$. With $3126P + 2682Q = 3298P + 2817Q$, and knowing that $|P| = 10151$, we have:

$$\mathbf{0} = 172P + 135Q = (172 + 135n)P$$
$$172 + 135n \equiv 0 \pmod{10151}$$
$$n \equiv 1277 \pmod{10151}$$

We can verify that $1277P = Q$, and we can then calculate $1277R = (8156, 1546)$ in order to find the shared secret on the example above.

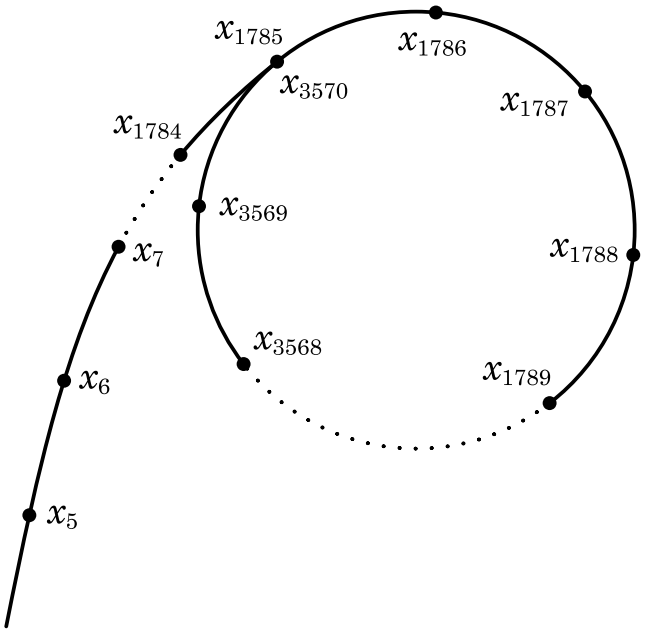


Figure 7: Diagram from [17], a visual explanation for the name (ρ) of the algorithm

Therefore, Pollard's ρ operates with the following steps:

- For finding k where $kP = Q$, first generate a random set of elements consisting of $M_i = x_iP + y_iQ$, then generate the initial point $A_0 = x_aP + y_aQ$
- With a specific criteria, do a random walk based on some property of A_n to make A_{n+1} (In our case, we used the ones digit)
- When a cycle is reached, we can tabulate all the coefficients in making the random walk to have a form like $aP + bQ = (a + c)P + (b + d)Q$, then

$$cP + dQ = \mathbf{0}$$
$$(c + kd)P = \mathbf{0}$$

And then k can be solved through c and d based on $|P|$.

Evaluation

Pollard's ρ algorithm on elliptic curve groups works on average with $\sqrt{\frac{\pi}{4}N}$ elliptic curve additions with $N = |P|$ [18]. The GNFS for finite fields has time complexity $\exp\left((64/9)^{1/3}(\ln p)^{1/3}(\ln \ln p)^{2/3}\right)$ in a prime field with order p. Assigning real numbers to these expressions, we can evaluate the current industry standards.

Diffie-Hellman in TLS 1.3

Transport Layer Security (TLS) is the protocol used in virtually all internet connections secured through cryptography [19], with the latest version being TLS 1.3. One important part of this protocol is Diffie-Hellman Key Exchange. We shall now examine the Diffie-Hellman methods it supports.

Finite Field Diffie-Hellman
The smallest finite field used by TLS for Diffie-Hellman is named ffdhe2048 [20], with the prime modulus defined as

$$p = 2^{2048} - 2^{1984} + \left(\lfloor 2^{1918} \cdot e \rfloor + 560316\right) \cdot 2^{64} - 1$$

The base is 2, with $|2| = (p-1)/2$. If we computed the expected running time for the general number field sieve to run, it requires about $1.00 \cdot 10^{35} \approx 2^{116}$ times than it would take for GNFS to run with order 3, providing about 116 bits of security. The original definition of ffdhe2048 gives a more conservative estimate that this provides 103 bits of security [21].

As this field uses a prime 2048 bits of size, each group element requires 2048 bits of storage.

Elliptic Curve Diffie-Hellman
The elliptic curve with the smallest element size supported by TLS is curve25519, using the prime $p = 2^{255} - 19$ as the field $\mathbb{F}_p$ the elliptic curve is over, and the curve $y^2 = x^3 + 486662x^2 + x$. The base point is $x = 9$, and the order of that point is $2^{252} + 27742317777372353535851937790883648493$. As per [22] the fastest known method to break DLP (a modified Pollard's ρ algorithm) takes $\sqrt{\frac{\pi}{4}N}$ group operations, this specific curve requires approximately $7.54 \cdot 10^{37} \approx 2^{126}$ operations to solve DLP, or providing approximately 126 bits of security.

As elliptic curve points have coordinates under the prime field $2^{255} - 19$, each coordinate value requires 255 bits of storage, therefore an entire point (both x and y coordinates) would take about 510 bits of storage.

Performance of group operations

Assume that multiplying two 256-bit integers has cost C. Multiplication of two 2048-bit integers thus will cost $64C$ as each 2048-bit integer has 8 256-bit digits and each of the eight digits from the first number needs to multiply with the eight from the second number [23, p. 398].

The story in elliptic curves is much more complicated. Curve25519 follows the form $By^2 = x^3 + Ax^2 + x$ called a Montgomery curve. All curves of that form can be transformed into the short Weierstrass form we used but not the other way around. Diffie-Hellman for curves in that form could be designed so that only the x-coordinate of each point in the process is needed, which simplifies the process by removing the need to compute y coordinates [24].

Under Montgomery arithmetic where only the x coordinates of curve points are involved, adding two curve points costs $3M + 2S + 3a + 3s$, where M, S, a, s are costs for multiplying two numbers, squaring a number, adding two numbers, subtracting two numbers in the field the curve is over respectively. Assuming that the cost for addition and subtraction is negligible compared to multiplication, and assuming that squaring has approximately equal cost as multiplying two numbers, the cost for adding two curve points is approximately $5M$. Note that the field is $2^{255} - 19$, so the cost of a multiplication M (for two 255-bit integers) can be considered as less than the cost of multiplying two 256-bit integers. So we have $M < C$.

Adding two curve points with curve25519 only costs $5M$, while multiplying in ffdhe2048 costs $64C$. (approximately 13x difference) As performing the group operation is the primary back-

bone behind Diffie-Hellman Key Exchange, this performance difference can have huge implications.

Comparison

The specific methods we have chosen to evaluate provide a general insight into the efficiencies of different methods of Diffie-Hellman Key Exchange.

Curve25519 only requires about 512 bits of storage for a full point, about 256 bits if only storing the x-coordinate, while in ffdhe2048, each element requires 2048 bits of storage, taking 8x as much storage than curve25519.

Adding two curve points in those groups compared to multiplying two finite field elements provide similar benefits in performance as well, with an approximate 13x difference in the number of operations required.

Both of these advantages can be seen from the fact that the Discrete Log Problem is much harder on elliptic curves than in finite fields in general, which we have shown above through comparing the General Number Field Sieve and Pollard's ρ algotithm. As a consequence, larger Finite Fields are required to provide the same level of security, which makes elliptic curves more efficient in comparison.

Conclusion

Elliptic Curve Cryptography offers a much better alternative to other existing cryptographic methods for establishing secrets through an insecure channel. This is partly because of the difficulty of the Discrete Log Problem for elliptic curves compared to other groups, which allows it to provide the same level of security while being more efficient. We theorize that using Elliptic Curves takes 8 times less storage than Finite Fields for Diffie-Hellman, and performs about 13 times faster.

Bibliography

[1] D. Warburton and S. Vinberg, "The 2021 TLS Telemetry Report." Accessed: Aug. 13, 2024. [Online]. Available: https://www.f5.com/labs/articles/threat-intelligence/the-2021-tls-telemetry-report

[2] K. Sako, "Public Key Cryptography," *Encyclopedia of Cryptography and Security*. Springer US, Boston, MA, pp. 487–488, 2005. doi: 10.1007/0-387-23483-7_331.

[3] PDClipart.org, "Safe Clip Art." Accessed: Oct. 11, 2024. [Online]. Available: https://commons.wikimedia.org/wiki/File:Safe_clip_art.png

[4] M. Just, "Diffie–Hellman Key Agreement," *Encyclopedia of Cryptography and Security*. Springer US, Boston, MA, p. 154–155, 2005. doi: 10.1007/0-387-23483-7_111.

[5] T. Shemanske, *Modern Cryptography and Elliptic Curves*, vol. 83. in The Student Mathematical Library, vol. 83. Providence, Rhode Island: American Mathematical Society, 2017. doi: 10.1090/stml/083.

[6] M. J. B. Robshaw, "One-Way Function," *Encyclopedia of Cryptography and Security*. Springer US, Boston, MA, pp. 887–888, 2011. doi: 10.1007/978-1-4419-5906-5_467.

[7] M. Friedl, N. Provos, and W. Simpson, “Diffie-Hellman Group Exchange for the Secure Shell (SSH) Transport Layer Protocol,” Mar. 2006, doi: 10.17487/RFC4419.

[8] L. Velvindron and M. D. Baushke, “Increase the Secure Shell Minimum Recommended Diffie-Hellman Modulus Size to 2048 Bits,” Dec. 2017. doi: 10.17487/RFC8270.

[9] L. C. Washington, *Elliptic Curves: Number Theory and Cryptography*, 2nd ed. in Discrete mathematics and its applications. Boca Raton, FL: Chapman & Hall/CRC, 2008.

[10] K. Nguyen, “Index Calculus,” *Encyclopedia of Cryptography and Security*. Springer US, Boston, MA, pp. 287–289, 2005. doi: 10.1007/0-387-23483-7_198.

[11] A. K. Lenstra, “L-Notation,” *Encyclopedia of Cryptography and Security*. Springer US, Boston, MA, p. 358–359, 2005. doi: 10.1007/0-387-23483-7_237.

[12] B. Kaliski, “Field,” *Encyclopedia of Cryptography and Security*. Springer US, Boston, MA, p. 458–459, 2011. doi: 10.1007/978-1-4419-5906-5_407.

[13] B. Kaliski, “Ring,” *Encyclopedia of Cryptography and Security*. Springer US, Boston, MA, pp. 1049–1050, 2011. doi: 10.1007/978-1-4419-5906-5_431.

[14] B. Kaliski, “Finite Field,” *Encyclopedia of Cryptography and Security*. Springer US, Boston, MA, p. 468–469, 2011. doi: 10.1007/978-1-4419-5906-5_409.

[15] J. H. Silverman and J. T. Tate, *Rational Points on Elliptic Curves*. in Undergraduate Texts in Mathematics. Cham: Springer International Publishing, 2015. doi: 10.1007/978-3-319-18588-0.

[16] SuperManu, “ECClines-2.” Accessed: Sep. 15, 2024. [Online]. Available: https://commons.wikimedia.org/wiki/File:ECClines-2.svg

[17] □□□, “Pollard rho cycle.” Accessed: Sep. 16, 2024. [Online]. Available: https://commons.wikimedia.org/wiki/File:Pollard_rho_cycle.svg

[18] D. J. Bernstein, T. Lange, and P. Schwabe, “On the Correct Use of the Negation Map in the Pollard rho Method,” in *Public Key Cryptography – PKC 2011*, D. Catalano, N. Fazio, R. Gennaro, and A. Nicolosi, Eds., Berlin, Heidelberg: Springer, 2011, pp. 128–146. doi: 10.1007/978-3-642-19379-8_8.

[19] C. Heinrich, “Transport Layer Security (TLS),” *Encyclopedia of Cryptography and Security*. Springer US, Boston, MA, pp. 1316–1317, 2011. doi: 10.1007/978-1-4419-5906-5_234.

[20] E. Rescorla, “The Transport Layer Security (TLS) Protocol Version 1.3,” Aug. 2018. doi: 10.17487/RFC8446.

[21] D. K. Gillmor, “Negotiated Finite Field Diffie-Hellman Ephemeral Parameters for Transport Layer Security (TLS),” Aug. 2016. doi: 10.17487/RFC7919.

[22] D. Hankerson and A. Menezes, “Elliptic Curve Discrete Logarithm Problem,” *Encyclopedia of Cryptography and Security*. Springer US, Boston, MA, pp. 397–400, 2011. doi: 10.1007/978-1-4419-5906-5_246.

[23] P. Pudlák, “The Complexity of Computations,” *Logical Foundations of Mathematics and Computational Complexity: A Gentle Introduction*. Springer International Publishing, Heidelberg, pp. 365–493, 2013. doi: 10.1007/978-3-319-00119-7_5.

[24] C. Costello and B. Smith, “Montgomery curves and their arithmetic,” *Journal of Cryptographic Engineering*, vol. 8, no. 3, pp. 227–240, Sep. 2018, doi: 10.1007/s13389-017-0157-6.