

Overview of Public-key Cryptography

Takashi Mori

[Summary]

Cryptography for securely exchanging confidential information over communication channels that may be subject to eavesdropping has a long history, and was also used for military and diplomatic applications during World War I. In encrypted communication, encryption and decryption are performed based on a key. However, with symmetric-key cryptography, where the encryption key and decryption key are identical, the challenge lies in how to securely share the common key between the sender and receiver. In other words, since it is not possible to transmit a common key over a communication channel that may be subject to eavesdropping, it is necessary to distribute the common key through a separate secure means. Public-key cryptography solves this challenge by preparing different keys for encryption and decryption, making the encryption key public as a public key, and keeping the decryption key secret as a private key. There are key exchange methods for securely sharing a secret key over a communication channel that may be subject to eavesdropping, and these too fall under the broader definition of public-key cryptography. Public-key cryptography is widely used in encrypted communications on the internet and has become an indispensable technology in our modern information society. This article explains the basic principles of RSA cryptography, Diffie-Hellman key exchange, and elliptic curve cryptography as current public-key cryptographic systems, and introduces post-quantum cryptography, which is difficult to break even using quantum computers, as next-generation public-key cryptography.

1 Introduction

Human society needs to exchange confidential information through secure communications. However, there is a risk of eavesdropping by third parties during actual communication. Wireless communication in particular poses the problem that interception by third parties is easy, but it is difficult for the legitimate sender and receiver to detect that interception has occurred. Consequently, encryption is needed to prevent exposure of secret information even if communications are intercepted by third parties. Encryption has a long history, and was also used in World War I for military and diplomatic purposes¹⁾. In other words, the purpose of cryptography is to securely transmit secret information over a communication channel that may be subject to eavesdropping.

For example, in classical cryptography (Figure 1), the sender (Alice) shifts each letter of the plaintext “hello” by 4 characters to produce the ciphertext “lipps,” which is then sent to the receiver (Bob). Even if the eavesdropper (Eve) intercepts the ciphertext “lipps,” she cannot make sense of it, whereas the receiver (Bob) can obtain the plaintext “hello” by shifting each letter of the received ciphertext “lipps” back by 4 characters. If we break this cipher down into “scheme” and “key,” the scheme is to shift each letter of the alphabet

by n characters, and the key is $n = 4$. Both the scheme and the key are required to break a cipher. However, in classical cryptography, knowing the scheme made it easy to break the cipher, so both the scheme and the key were kept secret. In the example above, if the scheme of shifting the alphabet by n characters is known, the cipher can always be broken by checking all 26 possible values of n .

On the other hand, in modern cryptography, the system is designed so that security is maintained even if the scheme is known, as long as the key remains unknown. Therefore, only the key is kept secret while the scheme is made public. It is believed that making the scheme public, rather than keeping it secret, allows it to be more widely studied and actually results in higher security.



Figure 1 Example of Classical Cryptography

In the classical cipher described above, the number of characters shifted during encryption, n , must equal the number of characters shifted back during decryption, n' (i.e., $n = n'$). A cipher in which the encryption key and the decryption key

are the same is called a symmetric-key cipher (or secret-key cipher). In symmetric-key cryptography, the challenge lies in how the sender and receiver can securely share the common key. In other words, since the common key cannot be sent over a communication channel that may be subject to eavesdropping, it is necessary to arrange a separate secure means of delivering the common key. Public-key cryptography solves this problem. As shown in Figure 2, the receiver (Bob) prepares two different keys for encryption and decryption: the encryption key is made public as the public key, while the decryption key is kept secret as the private key. The sender (Alice) encrypts the plaintext using the public key and sends the ciphertext, and the receiver (Bob) decrypts the ciphertext using the private key to obtain the plaintext. Since the public key used for encryption cannot decrypt the ciphertext, an eavesdropper (Eve) cannot obtain the plaintext even if she acquires both the public key and the ciphertext. The invention of public-key cryptography made it possible to securely transmit secret information using only a communication channel that may be subject to eavesdropping. This article explains the basics of the main public-key cryptographic schemes currently in widespread use, and also presents an overview of post-quantum cryptography (PQC), which represents the next generation of public-key cryptography.

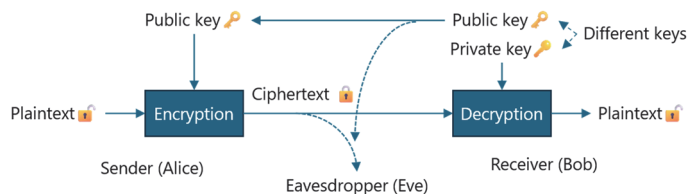


Figure 2 Public-key Cryptography Procedure

2 Features of Public-key Cryptography

The first challenge is the number of keys required for many-to-many encrypted communication. In symmetric-key cryptography, since sharing a key allows the ciphertext to be decrypted with that shared key, a different key is required for every possible combination of communicating parties. In contrast, with public-key cryptography, even if multiple senders share the same public key for encryption, security is maintained because the public key cannot be used to decrypt the ciphertext. For this reason, even when conducting many-to-many encrypted communication, the number of key pairs required is only equal to the number of receivers. For

example, in an 8-to-8 communication scenario (Figure 3), symmetric-key cryptography requires 28 keys, while public-key cryptography requires only 8 key pairs.

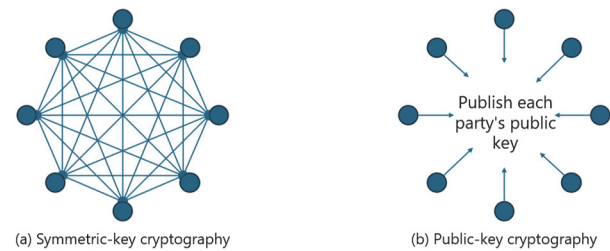


Figure 3 Number of Keys Required by 8-to-8 Cryptographic Communications

The second challenge is the encryption of large volumes of data. Although public-key cryptography has the advantage of not requiring sharing of a secret key and enabling many-to-many encrypted communication with ease, it has the drawback of requiring a large amount of computation for both encryption and decryption. Therefore, transmitting large volumes of data using public-key cryptography poses the challenge of placing a heavy computational load on the system. On the other hand, symmetric-key cryptography has the drawbacks of requiring secure sharing of a common key and needing a large number of common keys for many-to-many encrypted communication, but it has the advantage of requiring only a small amount of computation for both encryption and decryption. For this reason, encrypted communication over the internet commonly uses a method in which public-key cryptography is first used to share a key between the sender and receiver, and that key is then used to transmit data via symmetric-key cryptography. Since the amount of data involved in key sharing using public-key cryptography is small, the computational load does not pose a significant problem, and large volumes of data are transmitted efficiently via symmetric-key cryptography. At the same time, the sharing of the common key is carried out securely through public-key cryptography. This method thus cleverly avoids the drawbacks of both types of cryptography.

3 Current Public-key Cryptography²⁾

3.1 RSA Cryptography

RSA cryptography is a public-key cryptosystem published in 1978 by three researchers—Rivest, Shamir, and Adleman (RSA being their initials)—and is based on the difficulty of factoring large numbers into their prime factors. That is,

given two prime numbers p and q , it is easy to compute their product $n = p \times q$, but finding p and q from a given product n through prime factorization requires an enormous amount of computation.

Figure 4 shows the RSA cryptography procedure.

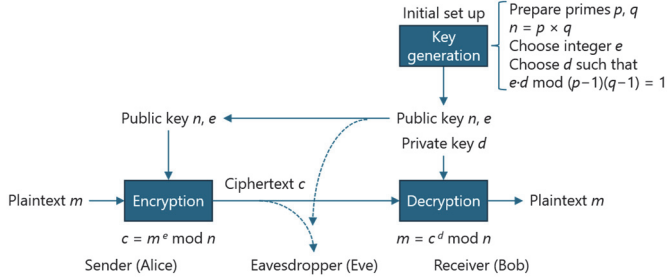


Figure 4 RSA Cryptography Procedure

First, the following initial setup is performed on the receiver side.

- Prepare two prime numbers p and q
- Compute the product of the primes $n = p \times q$
- Choose a positive integer e that is less than $(p-1)(q-1)$ and coprime to $(p-1)(q-1)$
- Choose d such that $e \cdot d \bmod (p-1)(q-1) = 1$ (where mod denotes the remainder of division)
- Make n and e public as the public key
- Keep d secret as the private key

This numerical example uses small numbers:

- Prepare two prime numbers $p = 3$, $q = 5$
- Compute the product of the primes $n = 3 \times 5 = 15$
- Choose a positive integer $e = 3$ that is less than $(3-1)(5-1) = 8$ and coprime to 8
- Choose $d = 3$ such that $e \cdot d \bmod (3-1)(5-1) = 1$ ($e \cdot d = 3 \times 3 = 9$, $9 \bmod 8 = 1$)
- Make $n = 15$, $e = 3$ public as the public key
- Keep $d = 3$ secret as the private key (In this numerical example e and d are equal, but in practice they would have different values.)

RSA cryptography makes use of the properties of exponentiation in mod n , specifically that raising an integer m to the power of $(p-1)(q-1) + 1$ modulo n returns the original m . For example, as shown in the numerical example in Table 1, raising any value of m to the power of $(p-1)(q-1) + 1 = 9$ modulo 15 returns the original m .

Table 1 Exponentiation Calculations for mod 15

m	Exponentiation								
	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	1	1	1	1
2	2	4	8	1	2	4	8	1	2
3	3	9	12	6	3	9	12	6	3
4	4	1	4	1	4	1	4	1	4
5	5	10	5	10	5	10	5	10	5
6	6	6	6	6	6	6	6	6	6
7	7	4	13	1	7	4	13	1	7
8	8	4	2	1	8	4	2	1	8
9	9	6	9	6	9	6	9	6	9
10	10	10	10	10	10	10	10	10	10
11	11	1	11	1	11	1	11	1	11
12	12	9	3	6	12	9	3	6	12
13	13	4	7	1	13	4	7	1	13
14	14	1	14	1	14	1	14	1	14

Let m denote the message to be transmitted. On the sender side, the message m is encrypted using the public key n and e according to the following equation, and the ciphertext c is transmitted.

$$c = m^e \bmod n \quad (1)$$

On the receiver side, the received ciphertext c is decrypted using the private key d according to the following equation.

$$m = c^d \bmod n \quad (2)$$

Through encryption and decryption, the message m is raised to the power of $e \cdot d$ modulo n . Since e and d are chosen during the initial setup such that $e \cdot d \bmod (p-1)(q-1) = 1$, and raising m to the power of $(p-1)(q-1) + 1$ modulo n returns the original m , the original message m is recovered through decryption. For example, as shown in the numerical example in Table 2, encrypting m by raising it to the power of $e = 3$ and decrypting the ciphertext by raising it to the power of $d = 3$ returns the original m .

Table 2 Encryption and Decryption Using Exponentiation

m	Encryption Exponentiation			Decryption Exponentiation		
	1	2	3	1	2	3
1	1	1	1	1	1	1
2	2	4	8	8	4	2
3	3	9	12	12	9	3
4	4	1	4	4	1	4
5	5	10	5	5	10	5
6	6	6	6	6	6	6
7	7	4	13	13	4	7
8	8	4	2	2	4	8
9	9	6	9	9	6	9
10	10	10	10	10	10	10
11	11	1	11	11	1	11
12	12	9	3	3	9	12
13	13	4	7	7	4	13
14	14	1	14	14	1	14

In RSA cryptography, n and e are made public, but decryption requires d . If p and q were known, d could be derived from $e \cdot d \bmod (p-1)(q-1) = 1$, and the cipher would be broken. Finding the prime numbers p and q such that $n = p \times q$ from a known n is a problem of prime factorization, and since prime factorization becomes extremely difficult when p and q are very large numbers, RSA cryptography is considered computationally unfeasible to break. However, as computing power improves, the speed of prime factorization also increases year by year, making it necessary to periodically increase the size of p and q . Note that in the initial setup on the receiver side, since the product $n = p \times q$ is computed after preparing the two prime numbers p and q , there is no need to perform prime factorization of large numbers.

The practical problems with RSA cryptography include issues arising from increasing key length and operational issues related to key generation³⁾. For increasing key length, while it is necessary to lengthen the key in response to improvements in computing performance in order to maintain security, processing long keys is difficult in environments with limited processing capacity, such as IC cards and embedded devices. The key generation issue is that RSA cryptography uses two large prime numbers, and if one of the primes is the same for different users, the private key can be derived relatively easily. The probability of different users randomly selecting the same prime number is negligibly small, but there is a possibility of using the same prime due to faults in random number generators or administrator errors. Even if one prime number is the same, the public key, which is the product of the two prime numbers, will differ, making it difficult to detect the duplication, and this is a problem.

3.2 Diffie-Hellman (DH) Key Exchange

DH key exchange is a key sharing method published in 1976 by Diffie and Hellman and is based on the difficulty of the discrete logarithm problem for large numbers. That is, given integers g and a and a prime number p , it is easy to compute the power $g^a \bmod p$. However, finding the integer a given the integer g , prime p , and the power $g^a \bmod p$ is a discrete logarithm problem that requires an enormous amount of computation.

Computation of $g^a \bmod p$ can be performed with a small

number of multiplications instead of performing a multiplications, as follows. Compute $x_2 = g^2 \bmod p$, $x_4 = x_2^2 \bmod p = g^4 \bmod p$, $x_8 = x_4^2 \bmod p = g^8 \bmod p$, $x_{16} = x_8^2 \bmod p = g^{16} \bmod p$, $\dots$ and then, for example, when seeking the 22nd power, compute $x_{16} \cdot x_4 \cdot x_2 = g^{16} \cdot g^4 \cdot g^2 \bmod p = g^{22} \bmod p$, which allows large powers to be computed efficiently.

Similarly, the logarithm problem of finding a when y_a and g are known in $y_a = g^a$ can be solved easily as follows. Compute $x_2 = g^2$, $x_4 = x_2^2 = g^4$, $x_8 = x_4^2 = g^8$, $x_{16} = x_8^2 = g^{16}$, $\dots$ and then efficiently find a by progressively narrowing down the range, for example as $x_{16} \cdot x_4 > y_a \geq x_{16}$. While the logarithm problem without $\bmod p$ can be solved efficiently in this way, the same approach cannot be applied to the discrete logarithm problem with $\bmod p$, making it difficult to find a .

Figure 5 shows the procedure for DH key exchange and is described below.

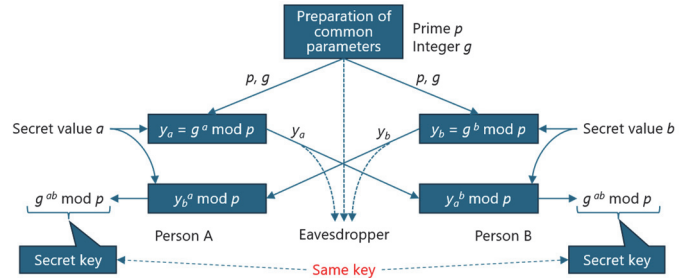


Figure 5 DH Key Exchange Procedure

- Prime p and integer g are made public as common parameters.
- Person A chooses a secret value a , computes $y_a = g^a \bmod p$, and makes it public.
- Person B chooses a secret value b , computes $y_b = g^b \bmod p$, and makes it public.
- Person A uses their own secret value a and Person B's public information y_b to compute the secret key $y_b^a \bmod p$.
- Person B uses their own secret value b and Person A's public information y_a to compute the secret key $y_a^b \bmod p$.
- Both parties' secret keys are equal to $g^{ab} \bmod p$.

To illustrate with a small numerical example:

- Prime $p = 23$ and integer $g = 9$ are made public.
- Person A's secret value is $a = 17$, and $y_a = 9^{17} \bmod 23 = 3$ is made public.
- Person B's secret value is $b = 5$, and $y_b = 9^5 \bmod 23 = 8$ is made public.

- Person A uses secret value $a = 17$ and Person B's public information $y_b = 8$ to compute the secret key $8^{17} \bmod 23 = 13$.
- Person B uses secret value $b = 5$ and Person A's public information $y_a = 3$ to compute the secret key $3^5 \bmod 23 = 13$.
- Both parties' secret key is 13.

In this way, both parties can share a secret key without ever transmitting it over a potentially eavesdropped channel. For an eavesdropper, since prime p , integer g , $y_a = g^a \bmod p$, and $y_b = g^b \bmod p$ are all public, if the value of a were known, the secret key could be derived via $y_b^a \bmod p$ and if the value of b were known, the secret key could be derived via $y_a^b \bmod p$. However, finding the value of a from $y_a = g^a \bmod p$, or finding the value of b from $y_b = g^b \bmod p$, is a discrete logarithm problem, and since making p a very large prime makes the discrete logarithm problem difficult to solve, breaking DH key exchange becomes computationally unfeasible. However, as computing power improves, the speed at which the discrete logarithm problem can be solved also increases year by year, so it is necessary to periodically increase p . Note that in the DH key exchange procedure, since the exponentiations $y_a = g^a \bmod p$ and $y_b = g^b \bmod p$ are computed after preparing the secret values a and b , there is no need to solve the discrete logarithm problem. RSA encryption described in the previous section can transmit arbitrary messages, so key sharing is also possible by sending a secret key as a message, making it usable for both data encryption and key sharing. In contrast, DH key exchange allows both parties to share some secret key, but cannot transmit arbitrary messages, making it a method dedicated solely to key sharing.

3.3 Elliptic Curve Cryptography (ECC)

ECC is a public-key cryptosystem independently proposed by Miller and Koblitz around 1985, and is based on the difficulty of the discrete logarithm problem on elliptic curves. That is, given a point G on an elliptic curve and a natural number s , it is easy to compute the elliptic curve multiplication $T = s \times G$. However, the elliptic curve discrete logarithm problem, which asks for the natural number s given a point G on an elliptic curve and the elliptic curve multiplication $T = s \times G$, requires an enormous amount of computation.

An elliptic curve is a curve expressed as $y^2 = x^3 + ax + b$,

and for example, the elliptic curve with $a = -1$ and $b = 1$ appears as shown in Figure 6. The concept of arithmetic operations on the elliptic curve is as follows: Let $G = (0.5, 0.790569)$ be a point on the elliptic curve. As shown in Figure 7, find the intersection point A of the tangent line at point G and the elliptic curve, then let B be the point obtained by inverting the sign of y at point A . This is defined as the elliptic curve addition and multiplication $B = G + G = 2 \times G$. Similarly, as shown in Figure 8, find the intersection point C of the line connecting point G and point B with the elliptic curve, then let D be the point obtained by inverting the sign of y at point C . This is defined as the elliptic curve addition and multiplication $D = B + G = 3 \times G$. Also, as shown in Figure 9, when there is no intersection point between the line connecting point A and point B and the elliptic curve, we set $A + B = O$, and define the subtraction on the elliptic curve as $B = O - A = -A$. The computation of the elliptic curve multiplication $T = s \times G$ can be performed with a smaller number of additions, as described below, rather than performing s additions. Similarly to the exponentiation calculation in the previous section, we compute $A_2 = G + G = 2 \times G$, $A_4 = A_2 + A_2 = 4 \times G$, $A_8 = A_4 + A_4 = 8 \times G$, $A_{16} = A_8 + A_8 = 16 \times G$, $\dots$, and when, for example, we wish to find the 22-fold multiple, we can efficiently compute multiplication by large numbers as $A_{16} + A_4 + A_2 = 16 \times G + 4 \times G + 2 \times G = 22 \times G$.

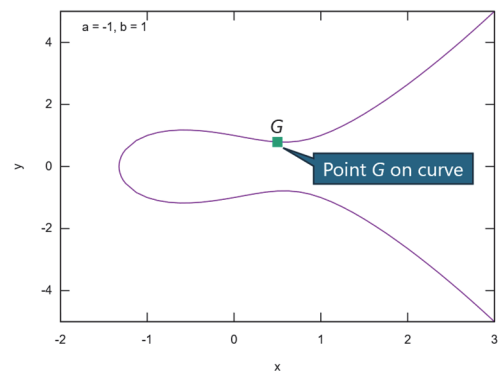


Figure 6 Example of Elliptic Curve

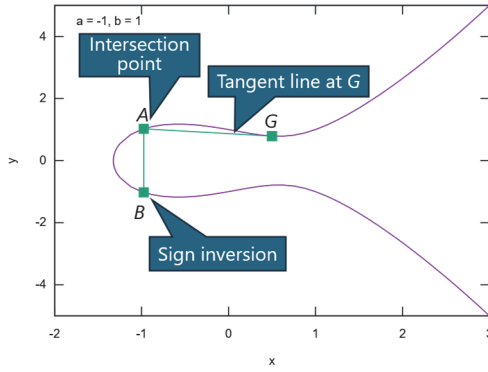


Figure 7 Example 1 of Addition on Elliptic Curve

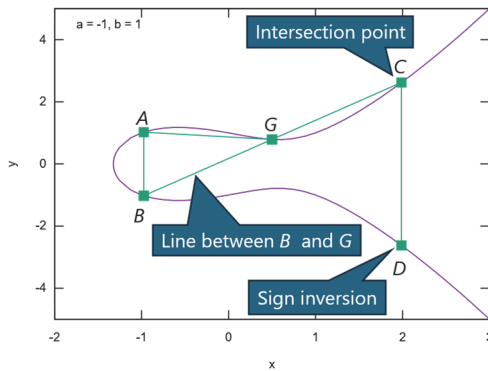


Figure 8 Example 2 of Addition on Elliptic Curve

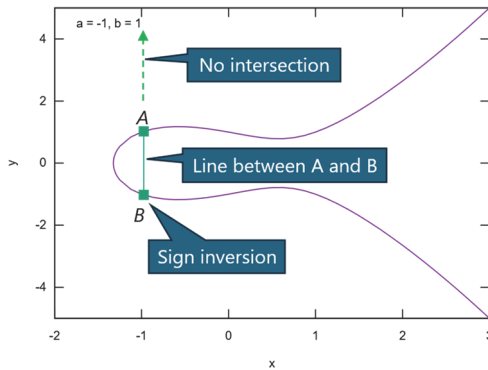


Figure 9 Example of Subtraction on Elliptic Curve

The discrete elliptic curve used in public-key cryptography is an integerized version of the elliptic curve described above. The concept is to let p be a prime number, and let x , y , a , and b be integers from 0 to $p-1$. The discrete elliptic curve is a discrete integer value expressed as $y^2 = x^3 + ax + b \pmod{p}$. Once a point G on the discrete elliptic curve is determined, if the natural number s is known, the multiplication $T = s \times G$ on the discrete elliptic curve can be computed efficiently in the same manner as described above. Conversely, the discrete logarithm problem of finding the natural number s from $T = s \times G$ is difficult to solve.

The procedure for ECC is explained under Figure 10.

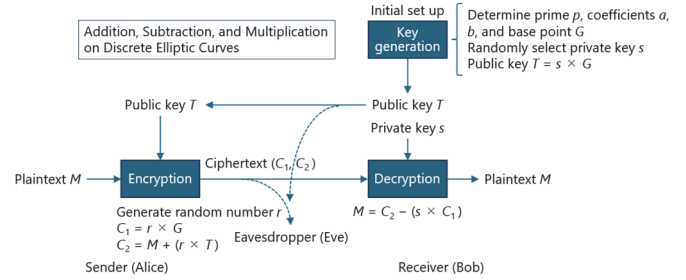


Figure 10 ECC Procedure

Key Generation:

- Publish the prime p , coefficients a , b , and base point G as common parameters.
- The receiver side randomly selects a private key s and publishes $T = s \times G$ as the public key.

Encryption:

- The sender side generates a random number r and computes $C_1 = r \times G$.
- To send the message M , compute $C_2 = M + (r \times T)$ and transmit (C_1, C_2) as the ciphertext.

Decryption:

- The receiver side computes $C_2 - (s \times C_1)$ using the ciphertext (C_1, C_2) and the private key s .
- As a result, $C_2 - (s \times C_1) = M + (r \times s \times G) - (s \times r \times G) = M$, and the message M is obtained.

A numerical example with small numbers is shown below:

Key Generation:

- Publish the prime $p = 263$, coefficients $a = 2$, $b = 3$, and base point $G = (200, 39)$ as common parameters.
- The receiver side selects the private key $s = 13$ and publishes $T = s \times G = (114, 104)$ as the public key.

Encryption:

- The sender side generates a random number $r = 35$ and computes $C_1 = r \times G = (219, 39)$.
- To send the message $M = (88, 84)$, compute $C_2 = M + (r \times T) = (250, 161)$ and send $(C_1, C_2) = ((219, 39), (250, 161))$ as the ciphertext.

Decryption:

- The receiver side computes $C_2 - (s \times C_1)$ using the ciphertext (C_1, C_2) and the private key $s = 13$.
- As a result, $(250, 161) - (13 \times (219, 39)) = (250, 161) - (201, 155) = (88, 84)$, and the message M is obtained.

The practical advantages of ECC include the benefits of short key lengths and operational aspects of key generation³⁾.

ECC can achieve equivalent security to RSA cryptography with a key length of approximately one-tenth of that required, making it suitable for use in environments with limited processing performance, such as IC cards and embedded devices. In ECC, a different natural number s should be selected for each user as the private key. If the same natural number s is selected by different users, the public key $T = s \times G$ will also be the same, making it easy to detect duplicates. This provides an operational advantage in key generation, because the problem of issuing weak keys is less likely.

3.4 Elliptic Curve Diffie-Hellman (ECDH) Key Exchange

ECDH key exchange is a method that performs DH key exchange using elliptic curves. Like ECC, it is based on the difficulty of the discrete logarithm problem on elliptic curves.

The procedure for ECDH key exchange is explained under Figure 11.

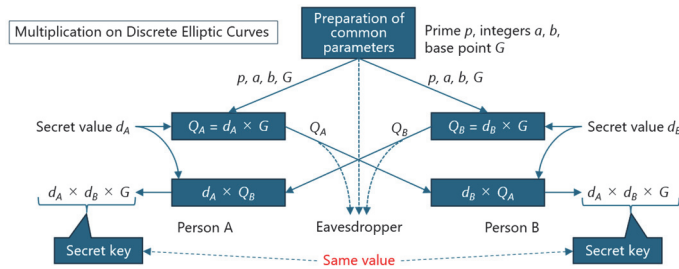


Figure 11 ECDH Key Exchange Procedure

- Publish the prime p , coefficients a , b , and base point G as common parameters.
- Person A decides on a secret value d_A and computes $Q_A = d_A \times G$ to publish.
- Person B decides on a secret value d_B and computes $Q_B = d_B \times G$ to publish.
- Person A computes the secret key $d_A \times Q_B$ using their own secret value d_A and Person B's public information Q_B .
- Person B computes the secret key $d_B \times Q_A$ using their own secret value d_B and Person A's public information Q_A .
- Both parties' secret keys are equal to $d_A \times d_B \times G$.

A numerical example with small numbers is shown below:

- Publish the prime $p = 263$, coefficients $a = 2$, $b = 3$, and base point $G = (200, 39)$ as common parameters.
- Person A decides on a secret value $d_A = 23$ and publishes $Q_A = d_A \times G = (198, 95)$.
- Person B decides on a secret value $d_B = 79$ and publishes $Q_B = d_B \times G = (129, 9)$.

- Person A computes the secret key $d_A \times Q_B = 23 \times (129, 9) = (133, 138)$ using their own secret value $d_A = 23$ and Person B's public information $Q_B = (129, 9)$.
- Person B computes the secret key $d_B \times Q_A = 79 \times (198, 95) = (133, 138)$ using their own secret value $d_B = 79$ and Person A's public information $Q_A = (198, 95)$.
- Both parties' secret keys are equal to $d_A \times d_B \times G = (133, 138)$.

In this way, both parties can share a secret key without transmitting it over a communication channel that may be subject to eavesdropping.

The ECC described in the previous section can transmit arbitrary messages, so key sharing is also possible by sending the secret key as a message, making it usable for both data encryption and key sharing. In contrast, ECDH key exchange allows both parties to share some secret key, but cannot transmit arbitrary messages, and is therefore a method dedicated exclusively to key sharing.

3.5 Strength of Public-key Cryptography

The security of the aforementioned public-key cryptography is based on the enormous computational effort required to solve the prime factorization and the discrete logarithm problem of large numbers. A comparison between the computational performance of the world's top supercomputer and the computational performance required to solve prime factorization and the discrete logarithm problem on elliptic curves within one year has been reported by CRYPTREC⁴⁾, and it would take an unrealistic period of over 1000 years to break the currently used 2048-bit RSA or 256-bit ECDH. However, since the computational performance of supercomputers improves year by year in accordance with Moore's Law, it is necessary to periodically increase the key length of cryptographic systems to raise the computational effort required for decryption. Conversely, by periodically increasing the key length, it is possible to maintain the security of public-key cryptography against the improvement in supercomputer computational performance that follows Moore's Law.

On the other hand, there is Shor's algorithm, which uses a quantum computer to efficiently solve prime factorization and the discrete logarithm problem. For this reason, although RSA cryptography and ECC are theoretically considered broken, breaking actual cryptographic systems requires a large-scale quantum computer, and current quantum

computers are nowhere near sufficient in scale, meaning that RSA cryptography and ECC have not actually been broken in practice. However, there is a technique known as a “Harvest Now, Decrypt Later attack” by which an attacker saves intercepted ciphertext and decrypts it at a future time when quantum computer performance has improved. This poses a threat for confidential data that does not become obsolete over time, and therefore countermeasures are needed in advance.

4 PQC

4.1 CRYSTALS-Kyber

Research and development of PQC that is difficult to break even using large-scale quantum computers is being actively pursued as successors to the current RSA cryptography and ECC. The US National Institute of Standards and Technology (NIST) launched a public solicitation for PQC standardization in 2016, and CRYSTALS-Kyber, based on lattice problems, was adopted as a key exchange algorithm and has been published as the standardization document FIPS 203⁵⁾. The lattice problem used in PQC is

$$\begin{cases} a_{11}s_1 + a_{12}s_2 + \cdots + a_{1n}s_n \approx t_1 \pmod{q} \\ a_{21}s_1 + a_{22}s_2 + \cdots + a_{2n}s_n \approx t_2 \pmod{q} \\ \vdots \\ a_{m1}s_1 + a_{m2}s_2 + \cdots + a_{mn}s_n \approx t_m \pmod{q} \end{cases} \quad (3)$$

the problem of finding $\mathbf{s} = (s_1, s_2, \dots, s_n)^T$ from $\mathbf{A} = (a_{ij})$ and $\mathbf{t} = (t_i)^T$ in the above equations of simultaneous linear approximation equations modulo prime q . When simultaneous equations use “=” without error, they can be solved efficiently. However, when the simultaneous equations use “ $\approx$ ” with error, they are difficult to solve.

The CRYSTALS-Kyber procedure⁶⁾ is shown under Figure 12.

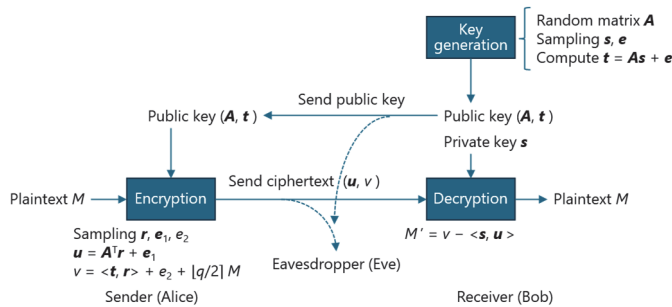


Figure 12 CRYSTALS-Kyber Procedure

Key Generation:

- Generate a matrix $\mathbf{A}$ randomly.
- Sample $\mathbf{s}, \mathbf{e}$ from a binomial distribution such that they are sufficiently small compared to q .
- Compute $\mathbf{t} = \mathbf{A}\mathbf{s} + \mathbf{e}$, and set $(\mathbf{A}, \mathbf{t})$ as the public key and $\mathbf{s}$ as the private key.

Encryption:

- Sample $\mathbf{r}, \mathbf{e}_1, \mathbf{e}_2$ such that they are sufficiently small compared to q .
- For plaintext M , compute $\mathbf{u} = \mathbf{A}^T \mathbf{r} + \mathbf{e}_1$, $v = \langle \mathbf{t}, \mathbf{r} \rangle + \mathbf{e}_2 + \lfloor q/2 \rfloor M$, and set $(\mathbf{u}, v)$ as the ciphertext. ($\lfloor a \rfloor$ denotes rounding to an integer, and $\langle \mathbf{t}, \mathbf{r} \rangle$ denotes the inner product of $\mathbf{t}$ and $\mathbf{r}$)

Decryption:

- Using the ciphertext $(\mathbf{u}, v)$ and private key $\mathbf{s}$, compute $M' = v - \langle \mathbf{s}, \mathbf{u} \rangle$; if $|M'|$ is less than $q/4$, the decrypted value is 0, otherwise it is 1.

4.2 Transition to PQC

In the transition to PQC, if the data protection period (period during which cryptographic protection is expected) is X years, the infrastructure migration period (period required to replace cryptographic processing) is Y years, and the period until cryptographic decryption by quantum technology is Z years, then as shown in Figure 13, the cryptography will be broken when $X + Y > Z$. Modern society uses public-key cryptography widely to protect various types of information, and the transition to PQC requires a long period (Y years) and considerable effort. While the likelihood of a quantum computer capable of breaking cryptography emerging in the near future is considered low, revolutionary advances in quantum technology could occur, making it difficult to estimate the period until cryptographic decryption by quantum technology (Z years). For this reason, the following responses are required: starting the transition to PQC as early as possible; implementing measures to shorten the infrastructure migration period (Y years) and the data protection period (X years); and monitoring developments in quantum technology⁷⁾.

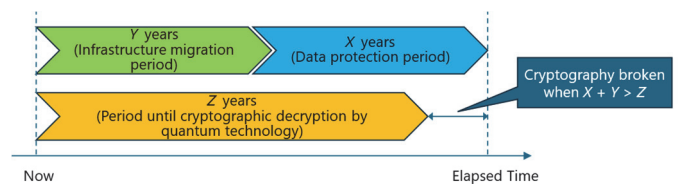


Figure 13 Mosca's Theorem

Social implementation of PQC has progressed in recent years. For example, Apple has announced the introduction of PQC into its iMessage app, which uses conventional ECC and PQC in combination to maintain security even if either one of the cryptographic systems is broken⁸⁾.

5 Conclusions

Cryptography is necessary to securely exchange confidential information over communication channels that may be subject to eavesdropping. However, symmetric-key cryptography requires sharing a common key between the sender and receiver, and the challenge lies in how to securely distribute this common key. Public-key cryptography solved this challenge by using different keys for encryption and decryption, making it possible to distribute the public key for encryption over a communication channel possibly subject to eavesdropping. Since public-key cryptography is also well-suited to many-to-many communication, it is widely used in encrypted communication on the current internet. This article explained the basic principles of RSA cryptography, DH key exchange, and ECC as current public-key cryptographic systems. However, it is known that these cryptographic systems can be efficiently broken when large-scale quantum computers are developed in the future, posing a threat to confidential data that does not become obsolete over time. For this reason, R&D into PQC is being conducted as a successor to the current RSA cryptography and ECC, and a method based on lattice problems has been adopted as a key exchange algorithm in the NIST PQC standard. Public-key cryptography is widely used in our modern information society, and while the infrastructure migration requires considerable effort, social implementation of PQC has already begun, and it is expected that the transition to PQC will continue going forward.

References

- 1) Shigeo Tsujii, "Development of Cryptography from for Military and Diplomacy to for Social Use," *Journal of IEICE*, vol. 106, no. 11, pp. 971–976, Nov. 2023 (in Japanese).
- 2) D.R. Stinson, *Cryptography Theory and Practice*, second edition, Chapman & Hall/CRC, 2002.
- 3) Takenobu Seito and Junji Shikata, "Recent Developments in Public-Key Cryptography: From RSA to Elliptic Curve Cryptography," *Kin'yu Kenkyu*, Institute for Monetary and Economic Studies, Bank of Japan, July 2013 (in Japanese).
<https://www.imes.boj.or.jp/research/papers/japanese/kk32-3-2.pdf>
- 4) CRYPTREC Report 2024, "Cryptographic Technology Evaluation Committee Report," Mar. 2025 (in Japanese).
- 5) National Institute of Standards and Technology, "Module lattice-based key-encapsulation mechanism standard," Aug. 2024.
<https://doi.org/10.6028/NIST.FIPS.203>
- 6) Masaya Yasuda, "Introduction to NIST-standardized Cryptography Based on Lattices," *Journal of IEICE*, vol. 106, no. 11, pp. 982–987, Nov. 2023 (in Japanese).
- 7) Tadahiko Ito, "Issues for the Transition to Post-quantum Cryptography," *Journal of IEICE*, vol. 106, no. 11, pp. 1026–1030, Nov. 2023 (in Japanese).
- 8) Apple Security Engineering and Architecture, "iMessage with PQ3: The new state of the art in quantum-secure messaging at scale," Feb. 2024.
<https://security.apple.com/blog/imessage-pq3/>

Author



Takashi Mori
Technology Planning Dept.
Advanced Research Laboratory

Publicly available