Post-quantum public-key cryptosystems and their problems

Koichiro Akiyama
Corporate Research and Development Center,
Toshiba Corporation

1 Introduction

Cryptosystems are a crucial technology for maintaining the security of IT and IoT systems. When sending confidential messages via the Internet, we should consider using a cryptosystem to encrypt them to keep secrecy throughout the communication channels. Cryptosystems are categorized as secret-key and public-key cryptosystems. Figure 1 illustrates how secret-key cryptosystems work. Sender Alice uses a shared key to encrypt a plaintext message into a ciphertext, which she sends to receiver Bob, who uses the same key to decrypt the ciphertext. The sender and receiver must securely exchange the shared key before communicating.

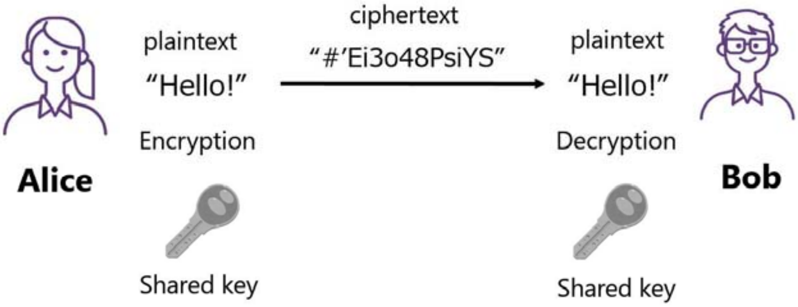

Figure 1: Encryption and decryption in a secret-key cryptosystem

One problem with secret-key cryptosystems is how to securely exchange the shared key. A trivial answer is for Alice and Bob to directly exchange it, but public-key cryptosystems provide a more elegant solution. Public-key cryptosystems utilize two keys, a public key and a private key, which are respectively used in the encryption and decryption phases. Figure 2 illustrates how public-key cryptosystems work. Bob (the receiver) generates the two keys and sends only his public key to Alice. Alice (the sender) is then able to use the public key to encrypt a shared key and to send it to Bob. Bob recovers the shared key by decrypting the ciphertext using his private key. The key point here is that Bob securely obtains the shared key, even when his public key is available on the Internet or other network. This works due to the computational complexity of a problem on which the security of the public-key cryptosystem depends. In the case of RSA, this underlying problem is integer factorization, which is well known as a computationally hard problem.

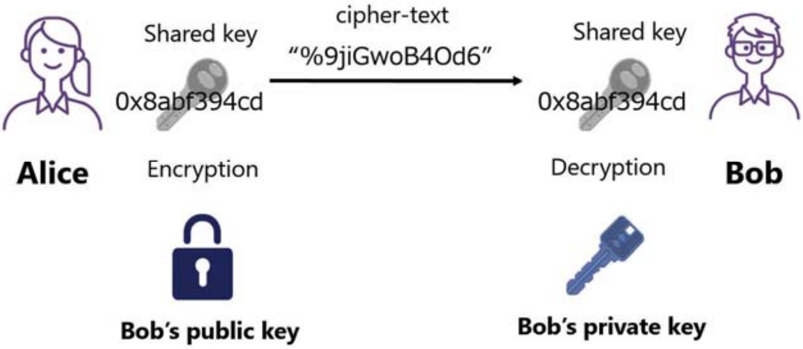

Figure 2: Encryption and decryption in a public-key cryptosystem

However, Shor presented algorithms for using a quantum computer to break RSA [2] and elliptic-curve cryptosystems [1] in a short time. There are predictions that quantum computers will break 2048-bit RSA with a probability of 1/2 by 2031 [3]. Research and standardization of post-quantum cryptosystems (PQCs) with the potential to resist attacks by quantum computers are beginning to evolve. This article surveys leading PQCs that are candidates for the NIST PQC standardization[1], and discusses their problems. We also describe potential technical directions for addressing those problems.

2 Preliminaries

This section describes the technical foundations of cryptosystems. Cryptosystems comprise keys and algorithms. An algorithm is also called a scheme or protocol. There are public and private keys, which work as described in the previous section. The algorithms perform encryption, decryption, and key generation, the latter of which usually takes parameters to produce a computationally hard problem that the system's security depends on.

In the case of RSA cryptosystems, security depends on an integer factorization problem. Even with state-of-the-art technology, a composite number with more than 2048 bits cannot be factored in real time[2]. Parameters are thus the bit length of composite number N and the bit length of N's prime factors p and q, which are set as 2048 and 1024, respectively. In the key generation algorithm, we first generate two distinct 1024-bit prime numbers p and q and calculate N by multiplying p and q. The composite number

[1] National Institute of Standards and Technology(NIST), where is US standardization institute, started the standardization process for PQC from December 2017.

[2] We assume composite numbers with only two distinct prime factors of the same bit length.

N is the public key. The private key is a positive integer d satisfying the relation

$$ed \equiv 1 \bmod (p-1)(q-1), \tag{1}$$

where e is an arbitrarily defined positive integer other than 1. The key generation algorithm calculates e from d by applying the extend Euclid's algorithm to Eq. (1). The encryption algorithm requires an integer m in the range $[0, N-1]$, which plaintext M is encoded into, and calculates

$$c = m^e \bmod N \ ,$$

where c is the ciphertext. Note that a 2048-bit N is sufficient to encode the shared key, the size of which is less than or equal to 256 bits. The decryption algorithm allows recovery of the plaintext m as

$$c^d = m^{ed} = m \bmod N \ .$$

Since the order of the irreducible residue class group $N/N\mathbb{Z}$ is $\Phi(N) = (p-1)(q-1)$, the relation $m^{ed} = m \bmod N$ holds by Fermat's theorem.

The above procedure is summarized as follows:

- Parameters
  1. N_{len}: the bit length of composite number N
  2. P_{len}: the bit length of odd primes p and q, which are factors of N.
- Keys
  1. Public key: a composite number N and positive integer e
  2. Private key: a positive integer d satisfying $ed \equiv 1 \bmod (p-1)(q-1)$
- Key generation
  1. Bob selects two primes p and q with p_{len} bits. Generating these prime numbers is easy with efficient primality testing algorithms such as the Miller–Rabin test or the Cohen–Lenstra test.
  2. Bob computes the composite number $N(=pq)$.
  3. Bob selects a positive integer e in the range $[2, (p-1)(q-1))$.
  4. Bob applies Euclid's algorithm to compute d by solving $ed \equiv 1 \bmod (p-1)(q-1)$.
- Encryption
  1. Alice encodes message M into the integer $m(\in \{0, \cdots, N-1\})$.
  2. Alice computes $c = m^e \bmod N$.
  3. Alice sends ciphertext c to Bob.
- Decryption
  1. Bob computes $c^d \equiv m^{ed} \equiv m \bmod N$.
  2. Bob recovers plaintext M from m.

The next section describes the parameters, keys, and algorithms for leading PQCs in a similar way as above.

3 Post-quantum cryptosystems

This section describes current leading PQCs.

3.1 Lattice-based cryptosystem

Lattice-based cryptosystems are well-studied PQC candidates. The security of these cryptosystems depends on the shortest or closest vector problems, which are NP-hard. The learning-with-error (LWE) problem is a widely used instance of these problems. Let q be a positive integer, $\mathbf{A}$ be a matrix in $GL_n(\mathbb{F}_q)$, and $\mathbf{e}, \mathbf{s}, \mathbf{b}$ be n-dimensional $\mathbb{F}_q$ vectors in the relation

$$\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}. \tag{2}$$

The LWE problem requires finding vectors $\mathbf{e}$ and $\mathbf{s}$ from given $\mathbf{A}$ and $\mathbf{b}$, where $\mathbf{e}$ is a small nonzero vector of elements sampled from a discrete Gaussian distribution. Elements in $\mathbb{F}_q$ are set as $\{0, 1, \cdots, q-1\}$ or $\{-(q-1)/2, \cdots, -1, 0, 1, \cdots, (q-1)/2\}$.[3] If vector $\mathbf{e}$ equals zero, this problem is equivalent to solving a linear equation that can be solved in $O(n^3)$. In other words, the LWE problem is equivalent to solving linear indeterminate equations (IEs) with $2n$ variables in n equations under the condition of finding a small vector of elements.

The following shows algorithms for the most basic lattice-based cryptosystems [4]. Here, the messages for encryption have only 1 bit (values 0 or 1).

- Parameters
  1. n: dimension of the lattice
  2. q: an odd prime used as the modulus that is sufficiently large for decryption
  3. α: a noise parameter
- Keys
  1. Public key: Matrix $\mathbf{A}(\in GL_n(\mathbb{F}_q))$ and vector $\mathbf{b}(\in \mathbb{F}_q^n)$
  2. Private key: Vector $\mathbf{s}$ satisfying $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}$, where $\mathbf{e}$ are small vectors, defined as those elements sampled from a discrete Gaussian distribution whose standard deviation and average are set as αq and 0 respectively.
- Key generation
  1. Bob selects a matrix $\mathbf{A}(\in GL_n(\mathbb{F}_q))$ uniformly at random.
  2. Bob selects a vector $\mathbf{s}(\in \mathbb{F}_q^n)$ uniformly at random.
  3. Bob selects a small vector (as defined above) $\mathbf{e}$.
  4. Bob computes vector $\mathbf{b}(= \mathbf{A}\mathbf{s} + \mathbf{e})$.
- Encryption
  1. Alice encodes message $M(\in \{0, 1\})$ to m by corresponding $M = 0$ to $m = 0$ and $M = 1$ to $m = (q+1)/2$

[3] When q is even, the elements are $\{1 - q/2, \cdots, -1, 0, 1, \cdots, q/2\}$ or $\{-q/2, \cdots, -1, 0, 1, \cdots, q/2 - 1\}$.

2. Alice creates a small vector $\mathbf{r}$.
3. Alice computes $\mathbf{C_1} = \mathbf{r}^T\mathbf{A}$.
4. Alice computes $C_2 = \mathbf{r}^T\mathbf{b} - m$. Note that $\mathbf{r}^T\mathbf{b}$ is a scalar.
5. Alice sends the ciphertext $(\mathbf{C_1}, C_2)$ to the receiver.

- Decryption

1. Bob computes $\mathbf{C_1 s} - C_2$, which equals $-\mathbf{r}^T\mathbf{e} + m$ as

$$\mathbf{C_1 s} - C_2 = \mathbf{r}^T\mathbf{A s} - \mathbf{r}^T\mathbf{b} + m = \mathbf{r}^T(\mathbf{A s} - \mathbf{b}) + m = -\mathbf{r}^T\mathbf{e} + m.$$

2. Bob decodes $M = 0$ or $M = 1$ depending on whether c belongs to the set $\{-(q+1)/4 < c < (q+1)/4\}$. See Figure 3.

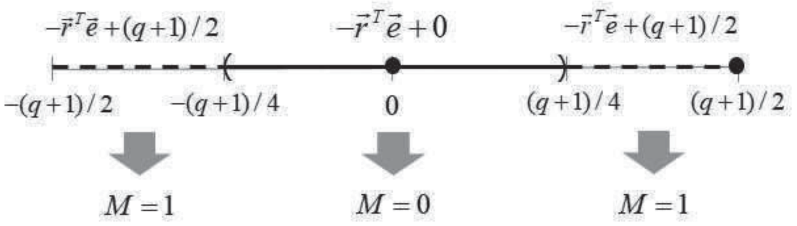

Figure 3: Decryption in lattice-based cryptosystems

Here, we assume

$$-(q+1)/4 < -\mathbf{r}^T\mathbf{e} < (q+1)/4$$

by setting parameters q and α.

To define parameters n, q, and α, lattice reduction algorithms such as LLL [5], BKZ [6], or BKZ 2.0 [7] should be considered; these are currently the most efficient algorithms for finding the shortest or closest vector belonging to a given lattice $\mathbf{A}$. NewHope [8], an NIST candidate for lattice-based cryptography, applies a method called the "2016 Estimate" to evaluate security in parameter estimation. Table 1 shows recommended secure parameters for satisfying 128-bit security [9].

Table 1: Performance in lattice-based cryptosystems

Parameters	Data size (bytes)			Process time (kilo-cycles)		
(n,q)	Public key	Private key	Ciphertext	Encryption	Decryption	Keygen
(512,12289)	928	869	1088	449	116	300

This table shows that public key sizes are much larger than in RSA, which requires 256 bytes.

3.2 Code-based cryptosystems

Code-based cryptosystems are designed using error-correcting codes capable of removing noise added to the communication channel. Their security depends on the Learning Parity with Noise (LPN) problem, which is related to decoding problem for linear codes.

Let χ be a distribution of $\mathbb{F}_2$ and $\mathbf{s}$ elements of $\mathbb{F}_2^k$. We then define a probabilistic function $O_{s,\chi}$ (called an "oracle") as follows. Oracle $O_{s,\chi}$ selects $\mathbf{a}$ from $\mathbb{F}_2^k$ uniformly at random and samples e from $\mathbb{F}_2$ according to distribution χ. Then, $O_{s,\chi}$ outputs $(\mathbf{a}, b)$ such that $b = \mathbf{s} \cdot \mathbf{a}^T + e$. The computational LPN problem LPN_χ requires a solution $\mathbf{s}$ from the $(\mathbf{a}, b)$s output by oracle $O_{s,\chi}$. If we limit n samples from the oracle $O_{s,\chi}$, we call this problem $LPN_{\chi,n}$, which is believed to be hard.

The $LPN_{\chi,n}$ problem is equivalent to solving the linear equation

$$\mathbf{sA} + \mathbf{e} = \mathbf{b},$$

where $\mathbf{A} = (\mathbf{a_1}^\mathbf{T}, \mathbf{a_2}^\mathbf{T}, \cdots, \mathbf{a_n}^\mathbf{T})$ and $\mathbf{b}$= $(b_1, b_2, \cdots, b_n)$. This is similar to the LWE problem described in section 3.1, the difference being that elements of $\mathbf{A}$, $\mathbf{e}$, and $\mathbf{s}$ belong to $\mathbb{F}_2$ instead of $\mathbb{F}_q$, where q is a positive integer more than one. LPN is thus a special case of LWE.

The following shows keys and algorithms for the McEliece code-based cryptosystem [10].

- Parameters
  1. t: minimum Hamming distance of the code
  2. k, n: size of the code generator matrix
- Keys
  1. Private key
     - $\mathbf{S}$: a regular matrix in $GL_k(\mathbb{F}_2)$
     - $\mathbf{P}$: an $n \times n$ permutation matrix that permutes n elements of vectors. This permutation matrix is defined as a square matrix that transforms any vector to a vector with its elements permutated, such as

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

     - Ψ: an error correcting algorithm corresponding to generator matrix $\mathbf{G}$
  2. Public key: a code generator matrix $\mathbf{G}'(\in M(n, k; \mathbb{F}_2))$ such that $\mathbf{G}' = \mathbf{SGP}$
- Key generation
  1. Bob selects a binary $(n \times k)$-linear code C that can correct t-bit errors. Code C must have an efficient decoding algorithm and generator matrix $\mathbf{G}$.
  2. Bob creates generator matrix $\mathbf{G}$ corresponding to code C.
  3. Bob selects regular matrix $\mathbf{S}(\in GL_k(\mathbb{F}_2))$ uniformly at random.

4. Bob selects an $n \times n$ permutation matrix $\mathbf{P}$ uniformly at random.
5. Bob computes $\mathbf{G}^{'}(= \mathbf{SGP})$.

- Encryption

1. Alice encodes k-bit message M into k-dimensional binary vector $\mathbf{m}$.
2. Alice selects n-dimensional binary vector $\mathbf{e}$ uniformly at random.
3. Alice computes ciphertext $\mathbf{c}$ as $\mathbf{mG}^{'} \oplus \mathbf{e}$.
4. Alice sends ciphertext $\mathbf{c}$ to Bob.

- Decryption

1. Bob computes $\mathbf{cP}^{-1}$, which equals $\mathbf{mSG} \oplus \mathbf{eP}^{-1}$.
2. Bob obtains $\mathbf{mS}$ by applying decoding algorithm $\Psi()$ to $\mathbf{cP}^{-1}$. Notice that the Hamming weight of $\mathbf{eP}^{-1}$ equals that of $\mathbf{e}$.
3. Bob computes $\mathbf{m}$ by $\mathbf{m} = (\mathbf{mS})\mathbf{S}^{-1}$.
4. Bob recovers the original message M from $\mathbf{m}$.

Parameters n, k, t are defined to be resistant against the following known attack algorithms.

The lattice reduction algorithms described in section 3.1 remain practical, since the LPN problem is a special case of the LWE problem. The BKW algorithm is particularly applicable to the LPN problem. The BKW algorithm aims to detect s_i (element i of $\mathbf{s}$) from given samples $(\mathbf{a}, b)$. Let $\mathbf{u_1}$ be a unit vector $(1, 0, 0, \cdots, 0)$. Given many $(\mathbf{u_1}, b)$ samples, we can detect variable s_1 from $b = s_1 + e$ by a majority decision, since many e values are expected to be 0 according to the distribution χ. We can collect many such samples by using linear algebra for translation [11]. Levieil and Fouque [12] further improved this attack.

Therefore, these parameters in code-based cryptosystems tends to be larger than those in lattice cryptosystems. Table 2 shows the parameters and performance of the Classic McEliece algorithm, which is a candidate for NIST standardization [13].

Table 2: Performance of code-based cryptosystems

Parameters	Data size (bytes)			Process time (kilo-cycles)		
(k,n,t)	Public key	Private key	Ciphertext	Encryption	Decryption	Keygen
(12,3488,64)	261,120	6,452	128	47	138	208,146

3.3 Multivariate cryptosystems

Multivariate cryptosystem security depends on the hardness of solving nonlinear multivariate equations on a finite field $\mathbb{F}_q$, such as

$$\begin{array}{rcl} f_1(x_1, x_2, \cdots, x_n) & = & c_1 \\ f_2(x_1, x_2, \cdots, x_n) & = & c_2 \\ \vdots & & \\ f_m(x_1, x_2, \cdots, x_n) & = & c_m \ , \end{array} \tag{3}$$

where $f_i(x_1, x_2, \cdots, x_n)$ $(i = 1, \cdots, m)$ are multivariate polynomials with n variables. In this section, $\mathbf{x}$ denotes variable vector $(x_1, x_2, \cdots, x_n)$ and $\mathscr{F}(\mathbf{x})$ denotes multivariate equations as in Equation (3). Solving nonlinear multivariate equations over $\mathbb{F}_2$ is a well-known NP-hard problem. The multivariate cryptosystem employs a quadratic polynomial as the multivariate polynomial and binary field $\mathbb{F}_2$ as the finite field. During encryption, n-bit plaintext is embedded in vector $(M_1, M_2, \cdots, M_n)$ in $\mathbb{F}_2^n$ and substituted into vector $(x_1, x_2, \cdots, x_n)$. We then obtain ciphertext $(c_1, c_2, \cdots, c_n)$ as the result of the substitution according to the equations in Equation (3), where the public key is multivariate polynomials $f_i(\mathbf{x})$ $(i = 1, \cdots, m)$. Recovering the plaintext by solving these equations is generally NP-hard. If we can take any multivariate equations $\mathscr{F}(\mathbf{x})$ as a public key, this cryptosystem is secure by NP-hardness. However, note that we cannot decode the ciphertext without its private key. In multivariate cryptosystems, this private key consists of multivariate equations $\mathscr{F}^{'}(\mathbf{x})$, which are easy to solve, and two matrices $\mathbf{S}(\in GL_m(\mathbb{F}_q))$ and $\mathbf{T}(\in GL_n(\mathbb{F}_q))$. The public key is then established as

$$\mathscr{F}(\mathbf{x}) = \mathbf{S} \circ \mathscr{F}^{'} \circ \mathbf{T}(\mathbf{x}).$$

The simplest example of constructing easy-to-solve equations is using a series of polynomials with sequentially fewer variables:

$$\begin{array}{lcl} f_1^{'}(x_1, x_2, \cdots, x_n) & = & c_1 \\ f_2^{'}(x_2, \cdots, x_n) & = & c_2 \\ \vdots & & \\ f_{n-1}^{'}(x_{n-1}, x_n) & = & c_{n-1} \\ f_n^{'}(x_n) & = & c_n \end{array} . \tag{4}$$

The last equation $f_n^{'}(x_n) = c_n$ is univariate. We easily obtain solutions s_n by applying polynomial factorization, whose computational complexity is polynomial. To obtain solutions s_{n-1}, we substitute s_n into x_n in the next-to-last equation $f_{n-1}^{'}(x_{n-1}, x_n) = c_{n-1}$, then obtain a univariate equation that can be similarly solved. By repeating this, we can find solutions $(s_1, s_2, \cdots, s_n)$ by solving in order from the bottom. Given several solutions s_i in each step i, we must branch and search for each solution.

The above is summarized as follows:

- Parameters

    1. n: the number of variables

2. m: the number of equations

- Keys
  1. Private key
     - $\mathscr{F}'(\mathbf{x})$: easy-to-solve multivariate equations
     - $\mathbf{S}$: an invertible matrix in $GL_m(\mathbb{F}_2)$
     - $\mathbf{T}$: an invertible matrix in $GL_n(\mathbb{F}_2)$
  2. Public key: Multivariate equations $\mathscr{F}(\mathbf{x})$ such that $\mathscr{F}(\mathbf{x}) = \mathbf{S} \circ \mathscr{F}' \circ \mathbf{T}(\mathbf{x})$
- Key generation
  1. Bob generates easy-to-solve multivariate equation $\mathscr{F}'(\mathbf{x})$.
  2. Bob creates an invertible $m \times m$ matrix S.
  3. Bob creates an invertible $n \times n$ matrix T.
  4. Bob computes $\mathscr{F}(\mathbf{x}) = S \circ \mathscr{F} \circ T(\mathbf{x})$.
- Encryption
  1. Alice encodes n-bit message M into n-dimensional binary vector $\mathbf{M}$.
  2. Alice substitutes n-dimensional binary vector $\mathbf{M}$ into public key $\mathscr{F}(\mathbf{x})$ and obtains the result $\mathbf{c}$ as a ciphertext. That is, $\mathbf{c} = \mathscr{F}(\mathbf{M})$.
  3. Alice sends the ciphertext $\mathbf{c}$ to Bob.
- Decryption
  1. Bob computes $\mathbf{c}' = \mathbf{T}^{-1}(\mathbf{c})$, which equals $\mathscr{F}'(S(\mathbf{x}))$.
  2. Bob solves the equation $\mathscr{F}'(S(\mathbf{x})) = \mathbf{c}'$ and obtains solution $\mathbf{s}$. Note that $S(\mathbf{M}) = \mathbf{s}$.
  3. Bob recovers the original message $\mathbf{M}$ by calculating $S^{-1}(\mathbf{s})$.

The size of public key $\mathscr{F}(\mathbf{x})$ can be defined as the total size of the coefficients including polynomials $\mathscr{F}(\mathbf{x})$. The polynomials include n equations, and each equation has $(n+1)(n+2)/2$ coefficients[4]. The size is then $n(n+1)(n+2)/2$. To avoid brute force attacks, we should make n greater than or equal to 128, so we can estimate the size as more than one megabit.

There are other methodologies for constructing easy-to-solve quadratic multivariate equations. The Matsumoto–Imai cryptosystem employs a univariate field equation whose solution is an element of extension field $\mathbb{F}_{2^n}$ [14]. Note that we can use an efficient polynomial factorization algorithm to solve this univariate field equation in polynomial time. Required solutions $(s_1, \cdots, s_n)$ can be mapped to an element of $\mathbb{F}_{2^n}$ as

$$\Psi : (s_1, \cdots, s_n) \longmapsto \sum_{i=0}^{n-1} s_i \beta^i,$$

[4] Each equation has $n(n+1)/2$ quadratic terms, n linear terms, and one constant term.

where $\{1, \beta, \beta^2, \cdots, \beta^{n-1}\}$ is a base of $\mathbb{F}_{2^n}$. When obtaining solution $\sum_{i=1}^{n} s_i\beta^i$ by solving field equation, we can recover $(s_1, \cdots, s_n)$ by Ψ^{-1}. In this case, matrices S and T are used to keep the field equation secret by linear translation of the variables. The Matsumoto–Imai cryptosystem and related systems are thus called hidden field-equation (HFE) cryptosystems.

However, Faugère suggested that Gröbner-basis calculations invalidate these linear translations [15]. Multivariate cryptosystems thus should increase the number of equations or variables, contributing to larger public keys.

4 Problems in post-quantum cryptosystems

As described in the previous section, a common problem in leading PQCs is their large public keys. This section discusses the reasons for this.

4.1 Linearity in public-key cryptosystems

RSA requires a 2048-bit public key, which is shorter than those for PQCs. This is because the underlying integer factorization problem is equivalent to finding a nontrivial integer solution to the equation $xy = N$. This is a particular problem related to nonlinear indeterminate equations(IEs) called the Diophantine problem, which is well known to be unsolvable in general. However, as described in section 1, there are known efficient quantum-computing algorithms for solving only $xy = N$ types of equations.

The underlying problems of lattice-based and code-based cryptosystems are linear IEs, which are NP-hard under the condition of finding the smallest solution. However, these problems are solvable in real time when we choose a low dimension. Due to well-known efficient algorithms such as LLL and BKZ mentioned in section 3.1, the cryptosystem must have dimensions of at least 512 to be secure. This contributes to larger public key sizes.

For a public key, multivariate cryptosystems use large equations constructed from easy-to-solve equations by applying linear translations S and T. Further, more equations or variables are needed for security, because linear translations can be invalidated by calculating Gröbner bases.

Therefore, a common problem among the leading PQCs is linearity in the computationally hard problems that these PQCs rely on. Reducing public key size would allow implementation on low-end devices, and also contribute to power savings by reducing processing costs.

4.2 An approach toward nonlinear cryptosystems

In 2006, Couveignes proposed a nonlinear key exchange scheme designed using isogeny maps of elliptic curves defined on a quadratic extension field $\mathbb{F}_q$ [16]. Security in this scheme depends on finding problems for the isogeny map and its kernel from two given distinct elliptic curves E and $\tilde{E}$. This scheme has been extended to public key cryptosystems [17] and signature scheme [18]. Supersingular isogeny key encapsulation is one such improved public key cryptosystem, and uses short public keys. However, the encryption and decryption processes take time due to calculations for multi-precision.

This section introduces a research approach that depends on solving IE problems as

IE problem: Find a solution $\mathbf{s} = (s_1, \cdots, s_n)$ $(n \geq 2)$ to the indeterminate equation $X(\mathbf{x}) = 0$ in ring R, where $\mathbf{x} = (x_1, \cdots, x_n)$ $(n \geq 2)$. All coefficients and variants are in ring R.

The IE problem has been proven to be unsolvable when ring R is a polynomial ring $\mathbb{F}_p[t]$. Indeterminate equation cryptosystems (IEC) are schemes that depend on problems related to IEs, where the public key is an indeterminate equation $X(\mathbf{x})$ and the private key is its solution $\mathbf{s} = (s_1, \cdots, s_n)$.

Giophantus$^+$ is the latest IEC depending on the IE problem on $R_q[t]$, where R_q is defined as $\mathbb{F}_q[t]/(t^n + 1)$. The IE problem on $R_q[t]$ is easy, so Giophantus$^+$ employs the problem of finding the smallest solution to IEs as a computationally hard problem. Here, "small" means all coefficients are in the range $R_\ell = \{0, 1, \cdots, \ell - 1\}$, where l is a small integer such as 4. R_ℓ is a subset of R_q.

The ciphertext presents a polynomial $c(\mathbf{x})$ in $R_q[\mathbf{x}]$ described as $c(\mathbf{x}) = m + X(\mathbf{x})r(\mathbf{x}) + \ell \cdot e(\mathbf{x})$. The first term m is an element of R_ℓ embedded with plaintext and the second term $X(\mathbf{x})r(\mathbf{x})$ is the product of polynomial $X(\mathbf{x})$, that is, the public key, and a random polynomial $r(\mathbf{x})$. The last term $\ell \cdot e(\mathbf{x})$ is a polynomial that can be eliminated after substituting the private key for $c(\mathbf{x})$. More details are described in [19, 20].

However, an attack called the linear algebra attack can target this scheme. This attack allows us to find the coefficients of polynomials $e(\mathbf{x})$ and $r(\mathbf{x})$, and thus m, by solving linear equations created by coefficient comparison of c and $m + X(\mathbf{x})r(\mathbf{x}) + \ell \cdot e(\mathbf{x})$. This is possible due to the linearity of the ciphertext.

5 Conclusion

This article surveyed some leading PQCs and described how a common problem is their large public key sizes. We discussed how the reason for this is linearity in the computationally hard problems that their security relies on. PQCs will no doubt play a central role in cyber-physical systems (CPS) that will be developed and will make life more convenient. CPS will require post-quantum public key cryptosystems that have low power requirements and are small enough to install in low-end devices such as sensors. It is therefore necessary to design PQCs whose security is based on inherently hard nonlinear problems such as solving Diophantine equations.

References

[1] P. W. Shor, "Polynomial time algorithms for discrete logarithms and factoring on a quantum computer", Proceedings of Algorithmic Number Theory, First International Symposium, pp. 289–289 (1994).

[2] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum", Computer, SIAM J. Comput., vol. 26, number 5, pp. 1484–1509 (1997).

[3] M. Mosca, "Cybersecurity in an era with quantum computers: will we be ready?", abstract for Invited Talk at the 5th International Conference on Quantum Cryptography, QCRYPT 2015, Tokyo, Japan, Oct 2, (2015).
`http://eprint.iacr.org/2015/1075`

[4] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography", Proceedings of the 37th Annual ACM Symposium on Theory of Computing, pp. 84–93 (2005).

[5] A. K. Lenstra, H. W. Lenstra, L. Lovász,"Factoring polynomials with rational coefficients", Mathematische Annalen, 261(4), pp. 515–534 (1982).

[6] C. P. Schnorr, M. Euchner, "Lattice basis reduction: Improved practical algorithms and solving subset sum problems", Mathematical programming, 66, pp.181-199 (1994).

[7] Y. Chen, P. Q. Nguyen, "BKZ 2.0: Better lattice security estimates", Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, vol. 7073, Lecture Notes in Computer Science, pp.1–20 (2011).

[8] E. Alkim, R. Avanzi, J. Bos, L. Ducas, A. de la Piedra, T. Pöppelmann, P. Schwabe, D. Stebila, "NewHope Algorithm Specifications, and Supporting Documentation",
`https://newhopecrypto.org/data/NewHope20171221.pdf`

[9] T. Pöppelmann, E. Alkim, R. Avanzi, J. Bos, L. Ducas, A. de la Piedra, P. Schwabe, D. Stebila, M. R. Albrecht, E. Osgeter, K. G. Paterson, G. Peer, N. P. Smart,"NewHope Algorithm Specifications, and Supporting Documentation",
`https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`

[10] R. J. McEliece. A public-key cryptosystem based on algebraic coding theory. Jet Propulsion Laboratory DSN Progress Report, 42-44:114-116, January and February 1978.
`https://ipnpr.jpl.nasa.gov/progressreport2/42-44/44N.PDF`

[11] A. Blum, A. Kalai, and H. Wasserman, "Noise-tolerant learning, the parity problem, and the statistical query model", J. ACM, 50(4), pp. 506–519 (2003).

[12] E. Levieil and P.-A. Fouque, "An improved LPN algorithm", In R. De Prisco and M. Yung, editors, Security and Cryptography for Networks, 5th International Conference, SCN 2006, Maiori, Italy, September 6-8, 2006, Proceedings, vol. 4116, Lecture Notes in Computer Science, pp. 348–359, Springer (2006).

[13] D. J. Bernstein, T. Chou, T. Lange, I. von Maurich, R. Misoczki, R. Niederhagen, E. Persichetti, C. Peters, P. Schwabe, N. Sendrier, J. Szefer, W. Wang,"Classic McEliece", NIST PQC Standardization Round 2 (2019) `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`

[14] T. Matsumoto and H. Imai, "Public Quadratic Polynomial-tuples for Efficient Signature Verification and Message Encryption", In EUROCRYPT '88, LNCS vol. 330, pp.419–453, Springer (1988).

[15] J. C. Faugère, A. Joux, "Algebraic cryptanalysis of hidden field equation (HFE) cryptosystems using Groebner bases," In CRYPTO '03 LNCS vol. 2729, pp.44–60, Springer (2003).

[16] J.M. Couveignes,"Hard homogeneous spaces", IACR Cryptology ePrint Archive, 2006:291 (2006).

[17] D. Jao, R. Azarderakhsh, M. Campagna, C. Costello, L. D. Feo, B. Hess, A. Jalali, B. Koziel, B. LaMacchia, P. Longa, M. Naehrig, J. Renes, V. Soukharev, D. Urbanik and G. Pereira,"SIKE", NIST PQC Standardization Round 2 (2019), `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`

[18] S.D. Galbraith, C. Petit, and J. Silva, "Identication protocols and signature schemes based on supersingular isogeny problems", In ASIACRYPT 2017, Part I, LNCS vol.10624, pp. 3-33, Springer (2017).

[19] K. Akiyama, Y. Ikematsu, Y. Wang, T. Takagi, "An examination for key recovery attack against a variation of indeterminate equation public-key cryptosystems", SCIS2019, 3B1-4 (2019) (in Japanese).

[20] Y. Ikematsu, Y. Wang, K. Akiyama, T. Takagi, "Experimental Analysis for Linear Algebraic Attack on a Variant of Indeterminate Equation Public-key Cryptosystems", SCIS2019, 3B1-3 (2019).

Cyber Security Technology Center
Corporate Research and Development Center
Toshiba Corporation
Kanagawa 212-8582
JAPAN
E-mail address: koichiro.akiyama@toshiba.co.jp