A Comprehensive Review of Post-Quantum Cryptography: Challenges and Advances

Seyed Mohammadreza Hosseini
M.Sc. graduate, Sharif University of Technology, Computer Engineering department

Hossein Pilaram
Assistant Professor, Sharif University of Technology, Electronics Research Institute

Abstract

One of the most crucial measures to maintain data security is the use of cryptography schemes and digital signatures built upon cryptographic algorithms. The resistance of cryptographic algorithms against conventional attacks is guaranteed by the computational difficulties and the immense amount of computation required to them. In the last decade, with the advances in quantum computing technology and the realization of quantum computers, which have higher computational power compared to conventional computers and can execute special kinds of algorithms (i.e., quantum algorithms), the security of many existing cryptographic algorithms has been questioned. The reason is that by using quantum computers and executing specific quantum algorithms through them, the computational difficulties of conventional cryptographic algorithms can be reduced, which makes it possible to overcome and break them in a relatively short period of time. Therefore, researchers began efforts to find new quantum-resistant cryptographic algorithms that would be impossible to break, even using quantum computers, in a short time. Such algorithms are called post-quantum cryptographic algorithms. In this article, we provide a comprehensive review of the challenges and vulnerabilities of different kinds of conventional cryptographic algorithms against quantum computers. Afterward, we review the latest cryptographic algorithms and standards that have been proposed to confront the threats posed by quantum computers. We present the classification of post-quantum cryptographic algorithms and digital signatures based on their technical specifications, provide examples of each category, and outline the strengths and weaknesses of each category.

Keywords**:** Post-Quantum, Quantum-Resistant, Cryptography, Data Security, Review

1. Introduction

Data security, whether during transmission or storage, is quite important to all individuals and organizations. According to the report published by the White House Office of Management and Budget on the budget of the United States government for 2024, the budget allocated to the Cybersecurity and Infrastructure Security Agency (CISA) for 2024 equals $3.1 billion, representing a $145 million increase compared to the previous year. The purpose of this increment is to make the United States cyberspace more resilient and defensible [1]. This example illustrates the increasing importance of cybersecurity, of which data security is one of its basic concepts [2].

The three core principles of data security are confidentiality, integrity, and availability. Various sorts of cryptographic algorithms and digital signatures are widely used to maintain all these principles [2], [3]. The resistance of cryptographic algorithms to conventional attacks (i.e., preventing the revelation of the plain text or the secret key) is ensured through the computational difficulties and the enormous amount of computation required to break them. In other words, the underlying principle of all cryptographic algorithms is that breaking them, even using the most powerful computers in the world, should take so long that the useful life of the encrypted data is over before the breaking [4]. In the article [4] presented by Joseph et al., the term "shelf-life" is used to refer to the useful life of data. According to the definition they provided, shelf-life is the time frame that the data must remain confidential and secure after transmission. Cryptographic algorithms and schemes are called "computationally secure" if the time required to break them, even using the most powerful supercomputers, exceeds the shelf-life of the data they are designed to protect [5].

Conventional cryptographic algorithms were all considered computationally secure according to the above definition when they were accepted as appropriate algorithms and considered as the foundation for cryptography and digital signature standards. However, a few decades after the introduction of these algorithms, the advent of quantum computing technology has led researchers to believe that attackers equipped with powerful quantum computers could break many of such algorithms in a short period of time (i.e., before the end of the shelf-life of encrypted data) [6]. Therefore, researchers began efforts to devise and design new cryptographic algorithms that would be resistant even to powerful quantum computers. These algorithms are called post-quantum cryptographic algorithms. Two of the basic principles in designing post-quantum cryptographic algorithms are that these algorithms must be resistant to both conventional (i.e., non-quantum) and quantum computers, and they must also be implementable and executable on conventional computers. In other words, it is assumed that attackers have access to both conventional and quantum computers, while people who try to protect their data are limited to using only conventional computers [5], [7], [8].

The National Institute of Standards and Technology (NIST) in the United States is one of the organizations working to identify and standardize suitable post-quantum cryptographic algorithms. In 2016, this institute initiated a competition inviting participants to submit cryptographic algorithms and digital signatures designed to be resistant to quantum computers. The NIST held this competition in four rounds. In each round, a selection of algorithms was made based on criteria such as security, temporal and hardware cost, and computational performance from the qualified algorithms and advanced to the next stage of the competition. Security was the most critical factor, defined as the resistance to being broken by all kinds of attacks that can be performed using conventional or quantum computers against cryptographic algorithms and digital signatures. The term "temporal cost" refers to the time required to execute the algorithm, and "hardware cost" means the hardware needed to implement the algorithm, including components such as the processor and memory. "Computational performance" also refers to the structural features of algorithms, such as simplicity in design and flexibility in implementation (i.e., the possibility of implementation on different platforms). Finally, at the end of the third round, four algorithms were selected to be standardized for worldwide practical encryption and digital signing in the future. Furthermore, the NIST competition continues into the fourth round, with new candidates being evaluated to identify at least one more post-quantum cryptographic algorithm for standardization and future use. The fourth round of the competition is still ongoing at the time of writing this article [8], [9], [10].

The major contributions of this study are listed as follows:

- This article explains and illustrates the vulnerabilities of conventional cryptographic algorithms and digital signatures against quantum computers, serving as a warning for users of such algorithms.
- This article provides a comprehensive review of various categories of post-quantum (i.e., quantum-resistant) cryptographic algorithms and digital signatures that can be utilized securely in the post-quantum era. Such information will be valuable to all individuals and companies involved in information technology and helps them to select the proper algorithms to secure their data and communications.
- The field of Post-Quantum Cryptography (PQC) still needs much research and development. By offering a comprehensive review of different categories of post-quantum cryptographic algorithms and digital signatures, along with their advantages, disadvantages, and limitations, this article provides worthwhile insights to researchers, motivating them for further research and improvements in this field.

This article is organized as follows: Section 2 briefly reviews conventional cryptographic algorithms and digital signatures, including their structural and functional information. Section 3 discusses the challenges and vulnerabilities posed by quantum computing to each category of conventional cryptographic algorithms. Section 4 provides a comprehensive review of various kinds of post-quantum cryptographic algorithms. Section 5 presents the algorithms selected by the NIST for encryption and digital signing in the post-quantum era. Section 6 provides valuable information about the implementation details of the post-quantum cryptographic algorithms and digital signatures selected by the NIST. Finally, Section 7 concludes this research and states possible ways to pursue this path.

2. Conventional Cryptography

In this article, the term "conventional cryptography" refers to the use of cryptographic algorithms and digital signature schemes designed and standardized in the last few decades. These algorithms and schemes were regarded safe before the advent of quantum computing, and even now, they are widely used in today's systems. In a classification, these algorithms are generally divided into two categories: symmetric-key and asymmetric-key (commonly referred to as public-key) cryptographic algorithms. In symmetric-key algorithms, there is only one key for both encryption and decryption, which is why they are called symmetric-key algorithms. Both the sender and receiver should have access to this key and must keep it confidential. One of the most well-known symmetric-key cryptographic algorithms is the Advanced Encryption Standard (AES) [11]. In the second category, each user possesses a pair of keys: a public key and a private key. Public keys are non-confidential and are shared with others, while private keys must remain confidential and be securely maintained by their respective owners. Asymmetric-key algorithms are designed such that a text encrypted with a user's public key can only be decrypted using the same user's private key. Conversely, if a text is encrypted with a user's private key, it can only be decrypted by the same user's public key. This is why such algorithms are called asymmetric-key algorithms. The corresponding private and public keys are different but mathematically associated to each other. It is not possible to derive one of them from the other. As two of the most celebrated asymmetric-key cryptographic algorithms, we can mention Rivest–Shamir–Adleman (RSA) public-key algorithm and the Elliptic-Curve Cryptography (ECC) scheme [11], [12].

Based on the features and structures of asymmetric-key algorithms, these algorithms are employed for both encryption and digital signing operations to maintain the confidentiality or integrity of the data. The meaning of confidentiality is to prevent unauthorized people from reading the data, and integrity refers to preventing unauthorized people from modifying the data. During the encryption

operation, the sender encrypts the data (referred to as plain text) using the recipient's public key (which is available to everyone) and sends the encrypted data (referred to as cipher text). When the recipient receives the encrypted data, he/she can decrypt it using his/her private key, which is accessible only to this user. Therefore, the data remains confidential while being transmitted, even through unsecured channels. During digital signing operation, the sender encrypts the data with his/her private key to generate his/her digital signature and sends the signature along with the original data to the recipients. When the recipients receive the signed data, they can verify its authenticity by decrypting the signature using the sender's public key and comparing its output with the received original data. Any mismatch detected indicates that the original text has been modified during the transmission. Hence, the integrity and authenticity of the data is protected during the transmission because unauthorized parties do not know the sender's private key, so they cannot alter the data and sign it again [8], [12], [13]. Fig. 1 and Fig. 2 show the details of encryption and digital signing operations using asymmetric-key cryptographic algorithms, respectively.

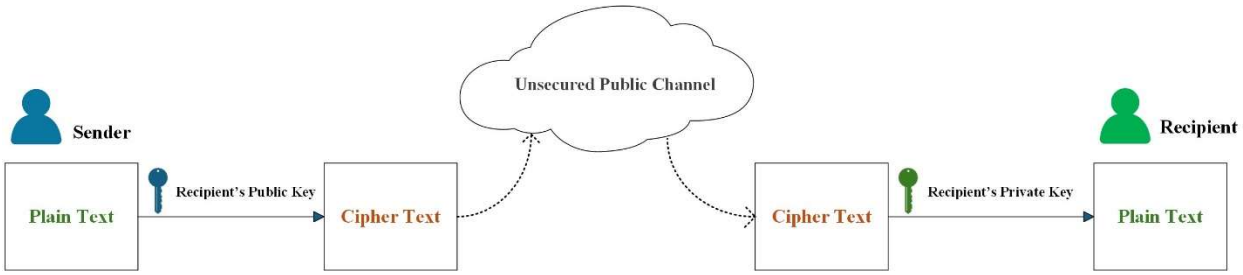


Fig. 1. encryption operation using asymmetric-key cryptographic algorithms

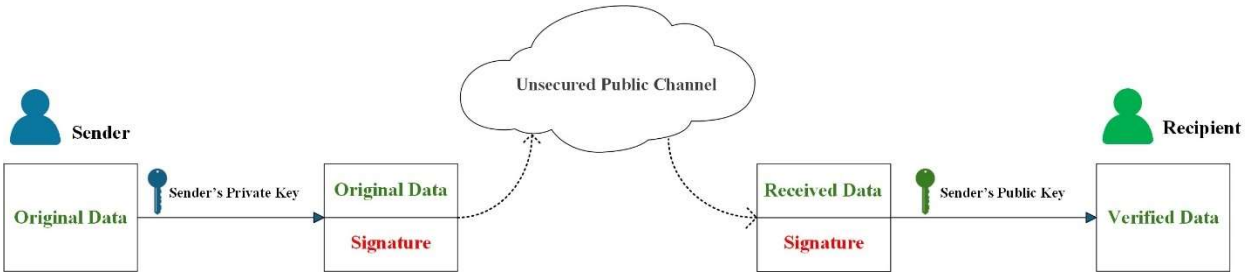


Fig. 2. digital signature operation using asymmetric-key cryptographic algorithms

Most symmetric-key cryptographic algorithms are based on operations such as XOR, rotation, and S-Box functions applied to the bits of the text. They aim to create as much confusion and diffusion as possible to make cryptanalysis attacks as difficult as possible. Therefore, in general, symmetric-

key cryptographic algorithms are faster than asymmetric ones and are often used to encrypt large text files, audio, video, or data streams [9], [11], [14]. On the other hand, most asymmetric-key cryptographic algorithms are designed based on difficult mathematical problems such as prime factorization and discrete logarithms, in which the time and computational costs to solve are based on exponential functions (i.e., with linear the increase in the key size, the time and computational cost increases exponentially). Due to the complex mathematical operations performed in these algorithms, most of these algorithms require a significant amount of time and computational resources for encryption and decryption. In other words, asymmetric-key algorithms are generally more costly to execute compared to symmetric-key algorithms [2], [10], [11].

Therefore, in most applications, with the assumption that each user knows his/her private and public keys and also has previously received the public keys of other users through secure channels, at the initiation of each communication session between two parties (e.g., user A and user B), an asymmetric-key cryptographic algorithm is used to securely transmit the secret key of symmetric-key cryptography. Then, a symmetric-key cryptographic algorithm is employed to encrypt the other data to be exchanged. In other words, if K_{pub-} and K_{pvt-A} are, respectively, the public and private keys of user A for asymmetric-key cryptography, and K_{sec} is the secret key of symmetric-key cryptography generated by user B (the initiator of the communication session), the operation $Encrypt(K_{pub-}, K_{sec}) \rightarrow Cipher$ is performed by user B, and subsequently, the operation $Decrypt(K_{pvt-A}, Cipher) \rightarrow K_{sec}$ is performed by user A, so that K_{sec} is delivered from user B to user A securely. Afterward, K_{sec} will be used as the key for symmetric-key cryptography between these two users. The process of secure key transmission using asymmetric-key cryptography is called "key encapsulation" operation. Also, asymmetric-key cryptographic algorithms used in key encapsulation are called Key Encapsulation Mechanisms (KEM) [15], [16], [17]. Fig. 3 shows the key encapsulation operation and also the symmetric-key cryptography, which is then performed using the transmitted secret key.

3. Challenges Facing Conventional Cryptography

As we mentioned in Section 1, with the advent of quantum computing, the security of some cryptographic algorithms and digital signatures was questioned. More precisely, the major challenges to conventional cryptographic algorithms arise from specific quantum algorithms introduced in recent decades. Quantum algorithms are algorithms that are designed according to the structure of quantum computers, which is based on quantum mechanics, and have the ability to solve some difficult mathematical problems (such as factorizing integer numbers into their prime factors) as well as tasks related to computer science (such as searching among unsorted data)

in a much shorter time than conventional (i.e., non-quantum) computers [5], [10], [18]. By using these algorithms on quantum computers, attacks leading to breaking conventional cryptographic algorithms can be carried out in a much shorter period of time. If this period of time is shorter than the useful life (shelf-life) of the encrypted data, data security will be endangered [4], [8].

In this section, we illustrate the security challenges posed by quantum computing and quantum algorithms for each kind of cryptographic algorithm that we explained earlier in Section 2. In Section 4, we explain the solutions provided by researchers to encounter these challenges and threats.

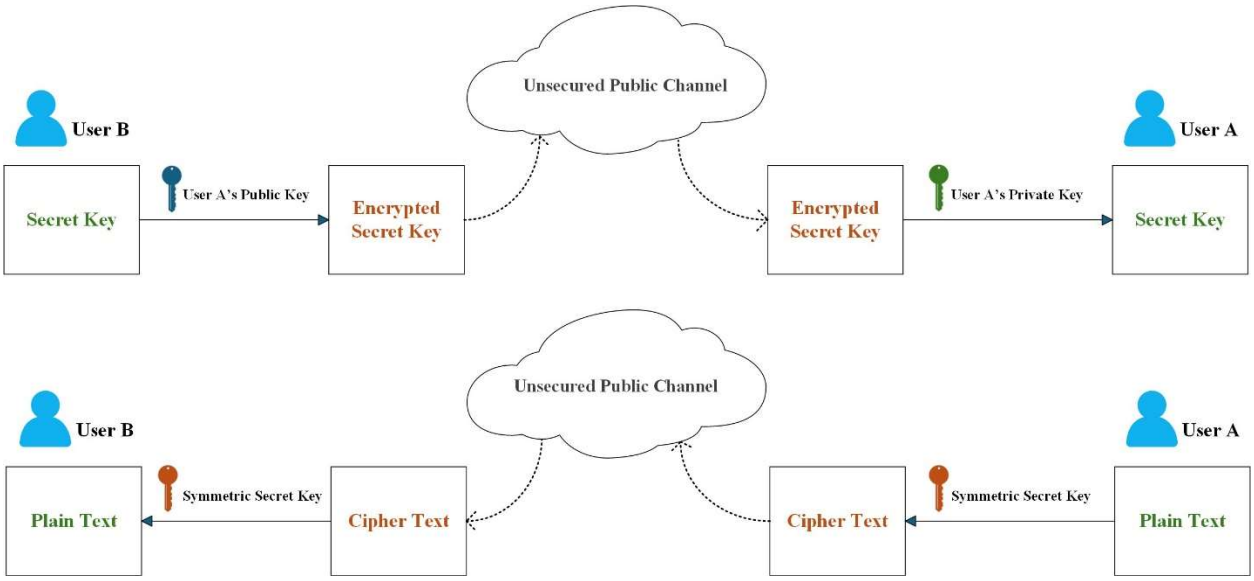


Fig. 3. key encapsulation operation and symmetric cryptography using the transmitted secret key

3.1 Challenges Facing Symmetric-Key Cryptographic Algorithms

In 1996, in research [19], Grover presented a quantum algorithm for searching among unsorted data, which reduced the time order of this type of search from $O(n)$ to $O(n^{1/2})$. By using this algorithm, which is one of the most important quantum algorithms, the duration of breaking symmetric-key cryptographic algorithms through the "brute force" attack is reduced to the square root of the original duration. [5], [10], [20]. The brute force attack is an attack in which the attacker tries all possible arrangements of bits to find the correct key [21]. To address this vulnerability and strengthen symmetric-key algorithms, it is necessary to double the size of the keys. For example, if in the past we used the AES algorithm with a key length of 128 bits, now we must use the AES with a key size of 256 bits to be as secure as before. Therefore, in the case of symmetric-key

cryptographic algorithms, the threats caused by quantum computing can be removed by increasing the key size [5], [10], [17], [22], [23].

3.2 Challenges Facing Hash Functions

Hash functions are one-way (irreversible) functions that take plain text as input, and then by performing operations such as XOR, rotation, and nonlinear functions on its bits, they hash it and generate a fixed-length output. MD5 and the family of SHA functions are some of the most famous hash functions [24]. Since these functions are often used in data security methods integrated with cryptographic algorithms and digital signatures, we review them in this section.

Since hash functions are one-way and irreversible, the input text can never be obtained from the output of the hash function, but because the size of the input text is larger than the output of the hash function, two or more input texts may have the same hash output. Attackers can exploit this characteristic and replace the original message with a fake message while the hash output is exactly the same for both texts. A widespread use case of these functions is to append the hash output of a message alongside it. This allows detection of any modifications to the message by comparing the newly generated hash output with the previous output that has been made from the original message. However, by exploiting the characteristic we explain above, two common attacks are performed against hash functions: pre-image and collision attacks [5], [24].

The definition of the pre-image attack is that the attacker tries to find an input text for a specific hash output so that if the hash function is run with the new input, the same specific output is produced. Therefore, the attacker can replace the original message with a fake message without any changes in the hash function output. In the collision attack, the attacker tries to find two plain texts with equal output of the hash function. Thus, these two texts can be replaced with each other without any changes in the output of the hash function [5], [24].

The quantum algorithms that were presented in the articles [25] and [26] based on Grover's algorithm are able to execute pre-image and collision attacks in $O(n^{1/2})$ and $O(n^{1/3})$ time orders, respectively. Here, n means the output size of the hash function. To strengthen hash functions and address the threats from quantum computers, hash functions with larger output sizes should be replaced [5], [10], [17], [27].

3.3 Challenges Facing Asymmetric-Key Cryptographic Algorithms

Before the concept of quantum computing emerged, it was believed that the time orders for solving the problems of factorization and discrete logarithms of integers were exponential functions of the number of digits in the input number. However, in articles [28] and [29], Peter Shor presented two quantum algorithms that can solve prime factorization and discrete logarithm problems of integers in polynomial time orders (instead of exponential time orders). This means that the time orders of these quantum algorithms are polynomial functions of the number of digits in the input numbers. These algorithms, which are among the most important quantum algorithms, caused the security of asymmetric-key cryptographic algorithms and digital signatures to be questioned because most of such algorithms are based on the above mathematical problems [8], [10]. For example, we can mention the asymmetric-key algorithm RSA, which is based on the prime factorization of integers, or the ElGamal and ECC asymmetric-key algorithms, which are designed based on discrete logarithms of integers [11]. Using Shor's algorithms in quantum computers, attackers can break these cryptographic algorithms in a short period of time and calculate people's private keys [5], [8], [10].

For instance, according to the functional details of the RSA algorithm, each user selects two large prime numbers, such as p and q, and generates his/her private and public keys from them. This user announces the multiplication of p and q (e.g., n) along with another random number (e.g., e) to everyone as his/her public key (i.e., $K_{pub} = (n, e)$). In this case, finding the two prime numbers p and q that have produced n takes a very long time and has an exponential time order on conventional (non-quantum) computers. However, this process requires a short time using Shor's algorithm on quantum computers, and its time order is a polynomial function. Therefore, the attacker (who knows n and e), by finding p and q, can calculate the user's private key from them [5], [11].

It should be noted that the security weakness of asymmetric-key cryptographic algorithms against quantum computing cannot be solved even by increasing their key sizes because the time orders of breaking these algorithms using Shor's algorithms will still be polynomial functions. Therefore, researchers presented other solutions that we discuss in the next section [5], [30], [31].

4. Post-Quantum Cryptographic Algorithms

As we explained in the previous section, unlike symmetric-key cryptographic algorithms, in the case of asymmetric-key algorithms, the vulnerability posed by quantum computers is not eliminated by increasing the key size. Therefore, researchers are looking for alternative approaches

to remove this vulnerability. In the last two decades, new asymmetric-key cryptographic algorithms have been proposed to be resistant to quantum computers. The basis of these algorithms, similar to conventional asymmetric-key algorithms, are problems (often mathematical problems) that are very difficult to solve, even using quantum algorithms on quantum computers [5], [10].

In this section, we review these algorithms, which are called post-quantum (or quantum-resistant) cryptographic algorithms and digital signatures. These algorithms, according to their technical specifications, are generally divided into five main categories, which are: lattice-based, code-based, hash-based, isogeny-based, and multivariate cryptography [5], [8], [10], [22]. In each category, algorithms for public-key cryptography, key encapsulation, and digital signatures are designed and presented to be resistant to quantum computing. In the following, we explain each category and provide examples.

Additionally, we describe the MPC-in-the-head category, a novel category of digital signature schemes, in this section. Due to its infancy, most renowned references in the field of post-quantum cryptography, such as [32], have not mentioned this category and its specifications or have not considered it a main category of post-quantum cryptographic algorithms and digital signatures. Describing this category and its characteristics, along with providing some instances of the known algorithms in this category, is one of the prominent contributions of this article and also one of its distinctions compared to the previous works.

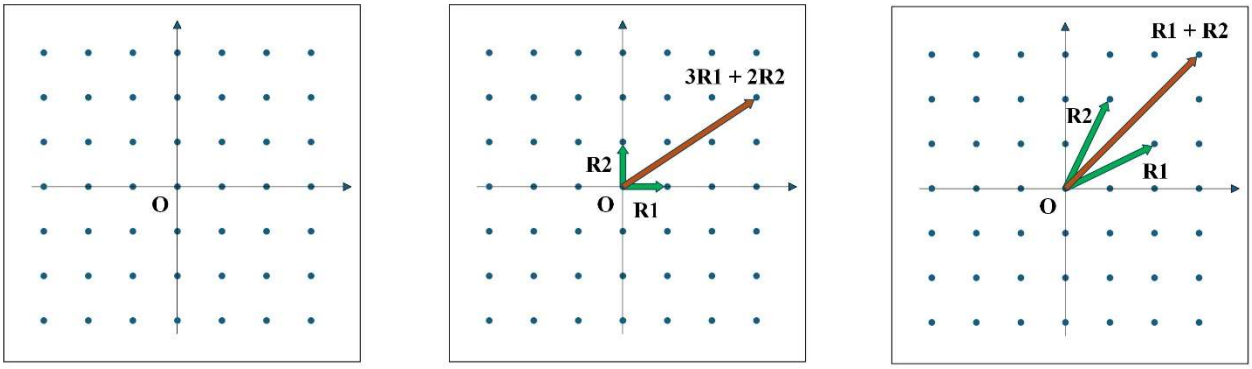
4.1 Lattice-Based Cryptography

A lattice is an infinite set of discrete points in an n-dimensional Euclidean space. Therefore, each point of an n-dimensional lattice is determined by n real numbers, which are the coordinates of that point. For example, the point $X = (x_1, x_2, \dots, x_n)$ in an n-dimensional Euclidean space can be a point in an n-dimensional lattice. Fig. 4a shows an example of 2-dimensional lattices. For each n-dimensional lattice, one or more n-membered sets of vectors can be found, which all of the nodes in the lattice can be reached from the zero point (origin) of the lattice through the linear combination of the vectors of each set. Each of such sets of vectors for a lattice is called the "basis" of that lattice, and the member vectors of each set are called "basis vectors". There may be more than one basis set for a lattice. Usually, to define and demonstrate a lattice, its basis is shown [33]. For example, a lattice like L is represented by the linear combination of its basis as follows:

$$L = \{\sum_{i=1}^{n} a_i v_i \,|a_i \in \mathbb{Z}\} \quad (1)$$

As we said above, the basis of a lattice is usually not unique, and there may be several bases for a lattice. In such a case, if a basis of a lattice contains short vectors (as shown by Fig. 4b), it is called a short basis (or good basis), and if it contains long vectors (as shown by Fig. 4c), it is called a long basis (or bad basis). In lattice-based cryptography, which is the largest category of post-quantum cryptographic algorithms, some mathematical problems related to lattices are used, which can be easily solved by having a short basis of the lattice. While on the other hand, having a long basis, it will be very difficult to solve them (especially when n is large enough). It is also very difficult to calculate the short basis from the long basis. Therefore, the short basis is used as the private key, and the long basis is used as the public key of the asymmetric cryptography. In this case, for decryption, the user who has the private key can easily solve the lattice problem. However, the users who only have the public key will have a very difficult and time-consuming task ahead, that has an exponential time order even using quantum computers [5], [33], [34].

Two of the most well-known lattice problems used in lattice-based cryptography are the Shortest Vector Problem (SVP) and the Closest Vector Problem (CVP). In the SVP, assuming we have one basis of the lattice, the shortest vector must be found that connects the zero point of the lattice to its nearest point. In the CVP, in addition to having one of the bases of the lattice, a random point in the space is also determined, and the closest point of the lattice to the determined point must be found. If we have the short basis of the lattice, these problems can be easily solved, but if we only have the long basis, solving these problems will be difficult and time-consuming [5], [33].



(a) A 2-dimensional lattice (b) An example of a short basis (c) An example of a long basis

Fig. 4. An example of a lattice and two sorts of its bases

Another fundamental lattice problem, which is the basis of some algorithms in this category, is the Short Integer Solution (SIS) problem. This problem is described as follows: Let $A \in \mathbb{Z}_q^{n\times m}$ be an $n \times m$ matrix with entries in $\mathbb{Z}_q$ (i.e., the ring of integers modulo q), that consists of m uniformly random vectors as its columns and $\beta > 0$ is a real number. This problem is shown as $SIS_{n,m,q,\beta}$,

and its goal is to find a non-zero integer vector like $x \in \mathbb{Z}^m$ such that: $Ax = 0\ (mod\ q)$ and $||x||_2 \leq \beta$ [35], [36]. There are a few well-known variants of the SIS problem, such as Inhomogeneous SIS (I-SIS), Ring-based SIS (RSIS), and Modulated SIS (MSIS). In recent decades, many lattice-based algorithms have been proposed based on these lattice problems [37].

One of the most widely used lattice problems, based on which many popular lattice-based algorithms have been designed and presented in recent years, is the Learning With Errors (LWE) problem [38]. The description of this algorithm is as follows: Let $\mathbb{Z}_q$ be the ring of integers modulo q, and $\mathbb{Z}_q^{n\times m}$ be the set of $n \times m$ matrices with entries from $\mathbb{Z}_q$. The input to the LWE problem consists of a uniformly random matrix like $A \in \mathbb{Z}_q^{m\times n}$ that $m \geq n$, a uniformly random secret vector like $s \in \mathbb{Z}_q^n$, and an error vector like $e = (e_1, e_2, \dots, e_m)$, in which each e_i has been sampled randomly according to an error distribution like $\mathcal{X}$. Given $A.s + e\ (mod\ q)$, the goal is to calculate the secret vector s, with a high probability (e.g., $p > 1 - \delta$) [33], [35], [37]. Similar to the SIS problem, the LWE problem has a few renowned variants, such as Ring-based LWE (RLWE) and Modulated LWE (MLWE) [37]. Many of the well-known and popular post-quantum cryptographic algorithms and digital signatures of the recent decade, such as CRYSTALS-Kyber [39] and CRYSTALS-Dilithium [40], are designed based on the LWE family of problems.

A noteworthy feature of lattice-based cryptographic algorithms is that their security levels are equal to the worst cases of the problems they are designed based on. While, most of the other algorithms, their security levels are equal to the average cases of their basic mathematical problems. Another advantage of this category of algorithms is their "algorithmic simplicity" and high "parallelization capability", which make them fast and convenient for execution by conventional computers. A notable drawback of these algorithms is the possible security vulnerabilities that may appear when implementing them on conventional computers [5], [8], [17], [34], [35].

One of the best-known algorithms in this category is the N-th-degree Truncated polynomial Ring Unit (NTRU) algorithm [41], based on which many newer algorithms were later built [9]. Algorithms in the NTRU family are mathematically based on the approximate Closest Vector Problem (appr-CVP) [42]. One of the distinctive advantages of the NTRU algorithm is that its complexity is $O(n^2)$, whereas many other public-key cryptographic algorithms have a complexity of $O(n^3)$. When comparing the same degree of security, the NTRU is 1.5 times quicker than the ECC algorithm. In the NTRU, key generation takes 300 times less time than the RSA algorithm, encryption takes 3 times less time, and decryption takes 30 times less time [9]. Specifically, the NTRU operations are based on objects in a truncated polynomial ring $R = \mathbb{Z}[x]/(x^N - 1)$. This means that all polynomials used in this algorithm have integer coefficients, and their degrees are

at most $N-1$ [43]. The functional details of the NTRU algorithm are summarized as follows [41], [43]:

- Firstly, the user, who owns the private and public keys, should select three integers such as N, p, and q, so that N is a prime number, q is considerably larger than p, and p and q are coprime. N is the polynomial degree bound, and p and q are small and large moduli, respectively. In this case, plain text messages are polynomials modulo p, and cipher text messages are polynomials modulo q.
- Then, the owner user should select the polynomials f and g in the ring R with coefficients in $[-(p-1)/2, (p-1)/2]$. The polynomial f must satisfy the additional requirement, that it have inverses modulo q and modulo p. These inverses can be calculated using Euclidean algorithm, and we denote them by f_p and f_q, that is $f.f_p = 1\ (mod\ p)$ and $f.f_q = 1\ (mod\ q)$.
- Key Generation: To generate the public key, the owner user should calculate $K_{pub} = p.(f_q.g)(mod\ q)$. The private key is the pair of (f, f_p), however in practice the user should also store g and keep it confidential.
- Encryption: To encrypt the message by a sender user, the plain text (e.g., M_p) is converted into a form of polynomial, in which the coefficients should be in the range of $[-(p-1)/2, (p-1)/2]$, and its degree should not be more than $N-1$. The message polynomial (i.e., M_p) can be translated in a binary representation. In addition, the sender user should select a blinding polynomial (e.g., r) in the ring R, which has equal positive and negative coefficients. The encrypted message M_c is computed as $M_c = (r.K_{pub}) + M_p\ (mod\ q)$.
- Decryption: If the sender user reveals the blinding polynomial (i.e., r), anybody knowing r will be able to compute the plain text (i.e., M_p) by evaluating $M_c - (r.K_{pub})$. Therefore, the sender user cannot send r to the recipient user (i.e., the owner of the keys). Hence, after receiving the message, the recipient user should obtain the plain text in two stages. Firstly, the recipient user should calculate $a = f.M_c\ (mod\ q)$ using the first element of the private key. Afterward, the user can utilize the second element of the private key and calculate the plain text as $M_p = f_p\ .a\ (mod\ p)$.

Some of the other renowned lattice-based cryptographic algorithms are NTRU-prime [44], SABER [45], and CRYSTALS -KYBER [39], which are designed for key encapsulation and public-key cryptography, and also FALCON [46], qTESLA [47], and CRYSTALS-DILITHIUM [40], which are presented for digital signature.

4.2 Code-Based Cryptography

With over 40 years of age, this group is considered the oldest category of post-quantum cryptographic algorithms, which are used as asymmetric-key cryptographic algorithms. The function of algorithms in this category is based on error detection and correction codes. It means the codes that are used to detect and even correct errors in transmitted data bits and are widely used in digital communication. The procedure of encryption in these algorithms is that on the sender's side, the data is first sent to an error correction code, and also, some random errors are added to it and then sent. On the receiver side, errors in the text are removed, and the text is decoded using another error correction code (associated with the sender's error correction code). In this case, the sender's error correction code is considered the public key, and the receiver's error correction code is considered the private key [5], [8], [10], [17].

The most reputable member of this category is the McEliece algorithm [48], which was presented in 1978 based on linear and binary Goppa code [49], which is an error-correcting code. Due to the fact that reversing the error-correcting code used in this algorithm is an NP-complete problem [50], the high security of this algorithm is proven. However, the excessively large public key has prevented this algorithm from being practical in the past years [5], [10], [17]. The way this algorithm works is as follows [5], [9], [48]:

- Firstly, the random number t and three matrices S, P, and G are determined. S is the "scrambler" matrix with dimensions $k \times k$ and should not be singular, and P is the "permutation" matrix with dimensions $n \times n$. The matrix G, with $k \times n$ dimensions, is the generator matrix for the selected linear binary Goppa code (which is an irreducible Goppa code). The matrix G can quickly decode the message and correct a maximum of t error bits. The number n equals the length of the selected Goppa code and is of the form $n = m^2$. The number k is determined in the form $k \geq n - tm$.
- Now, (G', t) is the public key, that $G' = S.G.P$, and (P, S, G) is the private key.
- On the sender's side, the message is divided into k-bit blocks for encryption. Then, a block of message like u is sent to the matrix G' to produce an n-bit output. After that, an n-bit random error vector with weight t (i.e., there are at most t bits of 1 in it), like z, is added to it to produce t bits of error in it. Therefore, the result is equal to $x = (u \cdot G') \oplus z$. Finally, x is sent to the recipient as the encrypted message (the cipher text).
- On the recipient's side, for decryption, at first, x is multiplied by the inverse of the matrix P, i.e., P^{-1}. So, we have $x' = x \cdot P^{-1} = \big((u \cdot G') \oplus z\big) \cdot P^{-1} = (u \cdot S \cdot G) \oplus (z \cdot P^{-1})$. Now, according to the fact that $z \cdot P^{-1}$ contains a maximum number of t bits of errors, by

sending the above result to the Goppa code decoder, we are be able to remove $z \cdot P^{-1}$ and calculate $u \cdot S$. Then, by calculating $(u \cdot S) \cdot S^{-1}$, u (i.e., the plain text) will be acquired.

As we said earlier, the security of this algorithm is guaranteed, but due to the huge size of the public key (e.g., about one megabyte) and the amount of additional data that is added to the plain text for encryption, that creates computational and communication overload, the algorithm has not been implemented and practical during the last decades. Although some efforts have been made to reduce the key size of this algorithm, they have failed and been broken [5], [9], [10]. Some of the other renowned algorithms in this category are BIKE [51], HQC [52] and [53], and RQC [54].

4.3 Hash-Based Cryptography

As we said in Section 3, the post-quantum security of hash functions can be ensured by increasing their output size. Therefore, most hash functions can be considered secure without any remarkable changes in their structure. Hash-based cryptography is often used to generate digital signatures and uses hash functions to generate the private and public keys used in digital signatures [5], [8], [10]. In this section, we introduce the most celebrated hash-based digital signatures that have been contributing and inspiring to other researchers to design newer algorithms.

In 1979, in the article [55], Lamport proposed a digital signature scheme based on hash functions. The function of Lamport digital signature is as follows [5], [55], [56]:

- Firstly, it is assumed that the signer user has a secure n-bit hash function and a random number generator.
- As the private key, the signer must generate n pairs of n-bit random numbers. Hence, the size of the private key is $2n^2$ bits.
- As the public key, the signer must generate the hash output of all the numbers in the private key, resulting in n pairs of n-bit hash outputs ($2n^2$ bits in total), which form the public key together. The signer can share the public key with everyone.
- To sign a message, the signer first generates an n-bit hash output of the message using the mentioned hash function. Then, for each bit in the message hash output, the pair of numbers corresponding to it is considered in the private key, and if the value of this bit is 0, the first number, and if the value of this bit is 1, the second number of the mentioned pair is selected and added to the signature. Therefore, one of each pair of numbers in the private key is selected, and a total of n n-bit numbers are placed together (i.e., the signature size equals n^2). The signature is sent to the recipient next to the plain text of the message.

- On the recipient's side, to verify the signature, firstly, the message's plain text is hashed by the same hash function, and an n-bit output is generated. Then, for each bit of this hash output, the corresponding pair of hashed numbers is considered in the public key. If the mentioned bit is equal to 0, the first number is selected, and if it is equal to 1, the second number is chosen from the hashed pair, and a total of n hashed numbers are selected.
- Then, the output of the hash of n numbers in the received signature is also generated. If all the n hashed numbers selected from the public key are equal to the hash output of the numbers in the received signature, that means the signature is correct and the message has not been modified during transmission. Otherwise, it means the message has been modified during transmission, and the signature is not valid.

It is noteworthy that the Lamport signature scheme is a one-time signature, and each pair of private and public keys should only be used once. If a pair of private and public keys is used more than once, its security is significantly reduced each time. For this reason, this digital signature algorithm is called Lamport's One-Time Signature (OTS) [5], [56], [57]. Fig. 5 shows Lamport's one-time digital signature.

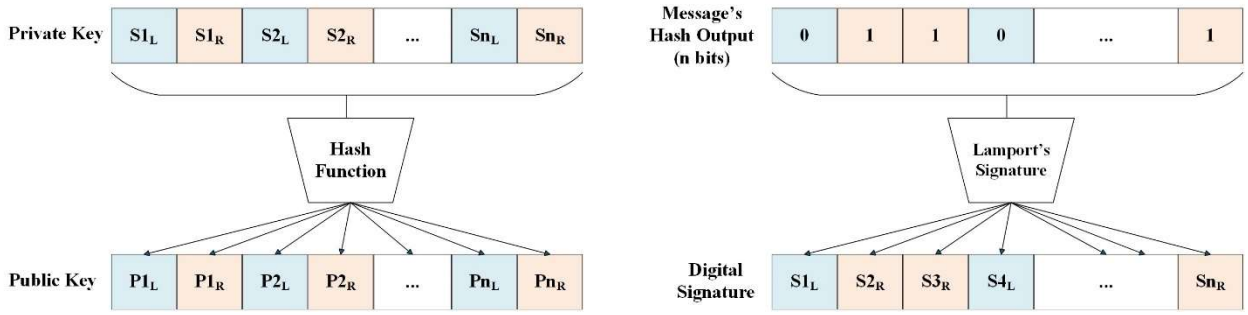


Fig. 5. Lamport's one-time digital signature scheme

Another prominent algorithm in this category was presented by Merkle in 1989 in research [57]. In that research, Merkle presented an algorithm for digital signature based on Lamport's OTS, which, unlike Lamport's algorithm, could be reused a limited number of times. Based on [57], [58], [59], [60], the function of this algorithm is as follows.

- To generate private and public keys, firstly, N pairs of private and public keys, such as (S_i, P_i), must be generated based on a one-time digital signature algorithm (such as Lamport's OTS). Note that N must be a power of 2, for example, $N = 2^n$. Then, a binary tree is formed, which has $n + 1$ levels and $2^{n+1} - 1$ nodes, and at its lowest level, the hash

outputs of N public keys (i.e., P_i's) are placed as leaves. For higher levels, the value of each node is equal to the hash output of the concatenation of its two children. For example, if $a_{1,0}$ is the node at level 1 and the first node from the left side, its children are $a_{0,0}$ and $a_{0,1}$, which are leaves at level zero. In this case, we have $a_{1,0} = Hash(a_{0,0}||a_{0,1})$. Such a tree is called a Merkle tree. Fig. 6 shows a Merkle tree for N=8 (i.e., with 4 levels) where the $H(P_i)$'s are the hash outputs of the public keys. The highest node (i.e., the root of this tree), which is the node $a_{n,0}$, can be shared with everyone as the overall public key (shown with PUB). The key PUB can be used for N times. In this algorithm, only the private keys (i.e., S_i's) are confidential, while the leaves and nodes of the tree are not confidential, and it is secure to share them, although it will significantly increase the size of the public key and transmitted data.

- On the sender's side, for encryption, firstly, one of the N pairs of the OTS private and public keys like (S_i, P_i) are selected. It should be noted that each pair should not be used more than one time so as not to reduce security. Then, the message's text is signed by the OTS with the selected private key (i.e., S_i). In addition to the plain text of the message and the generated one-time digital signature, it is necessary to include some additional information in the message so that the recipient can verify whether this signature is correct and valid. This additional information consists of the used public key (i.e., P_i) and some other intermediate nodes from the Merkle tree used so that the recipient can generate the path between the used public key (i.e., P_i) and the root node (i.e., PUB) with a hash function and make sure that the P_i belongs to the sender user.
- On the recipient's side, after receiving the message that contains the plain text, signature, and additional information, the recipient first checks if the plain text is validly signed with the received public key (i.e., the received P_i) to make sure that the text has not been modified during transmission. Then, as we said in the previous paragraph, using P_i and additional information (i.e., the values of some intermediate nodes of the tree), it tries to generate the path from P_i to PUB to ensure that the received P_i belongs to the sender user.

A Merkle signature can be used a limited number of times, which is equal to the number of pairs of public and private keys produced to form the Merkle tree. Therefore, this algorithm is also called Merkle's Few-Time Signature (FTS). In addition, Merkle's signature scheme is idiomatically "stateful". It is because of the fact that when you are using this algorithm, you should maintain the state of the used and unused pairs of public and private keys. Another weakness of this algorithm is the large public and private key and additional information that must be sent to the recipient along with the digital signature [5], [59], [60].

One of the most renowned studies that attempted to relieve the weaknesses of Merkle's signature scheme was conducted by Buchmann et al. They proposed an enhanced version of Merkle's signature, which is named eXtended Merkle's Signature Scheme (XMSS) [59]. They reduced the signature length by 25%, which resulted in the reduction of communication and computational load. However, the problem of statefulness still remained because XMSS is stateful as well.

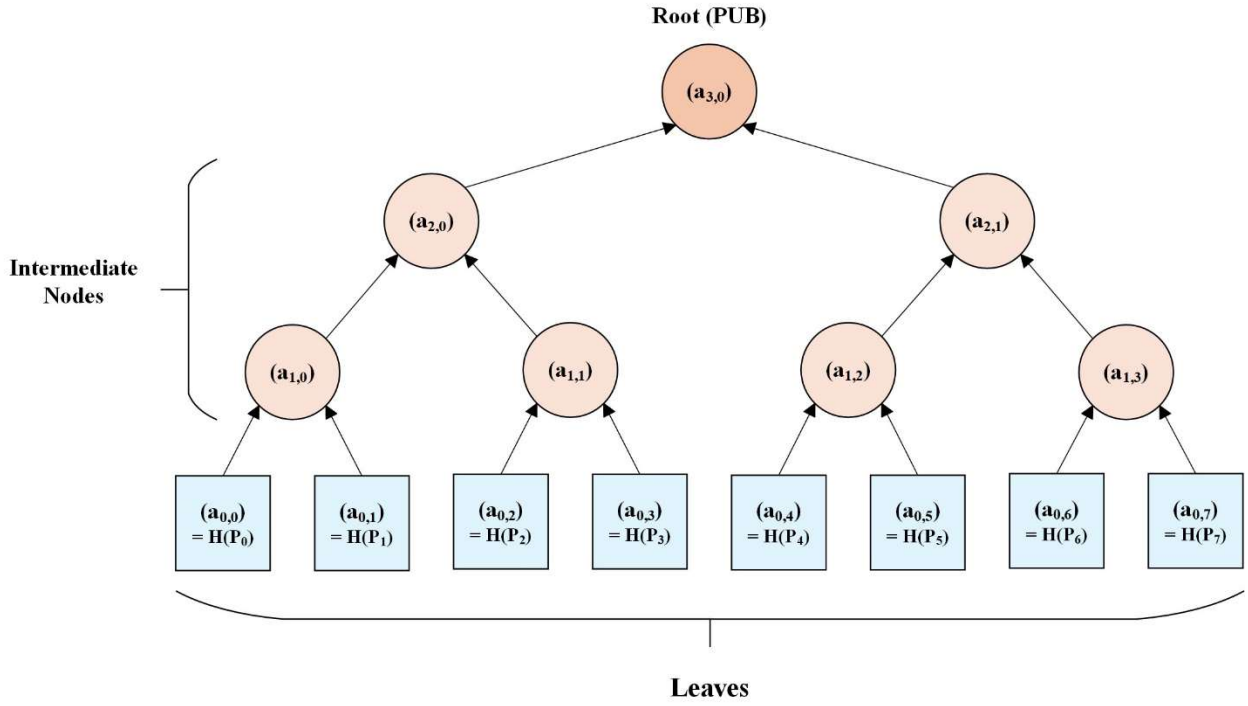


Fig. 6. Merkle tree for N=8 with 4 levels and 8 leaves

In the article [60], Bernstein et al. presented SPHINCS, one of the first celebrated stateless digital signature algorithms. Their strategy was to form a hypertree (i.e., a tree of trees) based on the Merkel tree idea and FTS algorithms. At the lowest level of this tree, as leaves, are few-time signature trees (i.e., FTS trees). To generate the few-time signatures, they used the HORST algorithm, which is an upgraded version of the old HORS algorithm [61] and has a smaller signature size, faster processing time, and higher security. To sign a message, each time the signer user randomly selects a signature (i.e., a pair of private and public keys) from one of the subtrees, each of them is an FTS, and using it, similar to what happens in the Merkel signature, signs the message. Although it is possible that a particular signature is accidentally used more than once (that is called a collision), due to the immense number of existing signatures, this probability is

insignificant. Even if this happens, due to the characteristics of the HORST few-time signature algorithm, the security of the signature does not reduce remarkably. Since using the SPHINCS signature does not require the signer user to maintain the state of the used and unused key pairs, this algorithm is called a "stateless" digital signature [60], [62].

The SPHINCS+ algorithm, which is one of the digital signature algorithms selected by the NIST, is an improved version of the SPHINCS [60] that was introduced in the article [62] by Bernstein et al. The improvements of SPHINCS+ compared to SPHINCS include: 1) Providing a new FTS algorithm called FORS, which has higher security, especially against collision attacks, compared to the HORST algorithm. 2) Using a newer method to randomly select the hypertree leaves that are used as keys in digital signatures. Together, these two changes increase the level of digital signature security and reduce the size of digital signature parameters (e.g., the signature and public key size), which means a remarkable reduction in the computational and communication load [62].

As some of the major advantages of hash-based digital signing algorithms, we can mention algorithmic simplicity and fast and convenient execution on conventional computers. On the other hand, some of the drawbacks of these algorithms are huge digital signatures as well as their security levels that are dependent on the security of utilized hash functions [5], [8], [62].

4.4 Multivariate Cryptography

It is proved that solving systems of multivariate (multi-variable) polynomial equations over finite fields are NP-hard problems. Therefore, these problems are suitable for designing cryptographic algorithms and digital signatures resistant to quantum computing because they are difficult and time-consuming to solve even when using quantum computers [5], [17], [63] This category of algorithms is useable for both public-key cryptography and digital signatures. The basis of a multivariate cryptographic algorithm is a system of multi-variable polynomial equations. If this system includes n variables and m polynomial equations with degree d, the size of the public key is equal to: $m \cdot \binom{n+d}{d}$. Usually, $d = 2$ is enough to ensure high security. In this case, the polynomials used are called quadratic polynomials, and most algorithms of this category use this type of polynomial. Other types of polynomials, such as cubic ones with $d = 3$, are also common [5], [17].

In a multivariate cryptographic algorithm, there is an easily invertible quadratic map like $F: F^n \rightarrow F^m$, and also there are two affine invertible linear maps like $S: F^m \rightarrow F^m$ and $T: F^n \rightarrow F^n$. In this case, the private key is a set of three matrices $SK = (S, F, T)$, and the public key is the multiplication of them i.e., $PK = (S \cdot F \cdot T)$. Here, the security of this algorithm is ensured by the

isomorphism problem, that is given the PK, the attacker should find two affine maps like S' and F' and an easily invertible quadratic map like F' such that $PK = S' \cdot F' \cdot T'$ [5], [17], [63].

For encryption usage, given the plain text $x \in F^n$, the sender user should calculate equation 2 to generate the cipher text $y \in F^m$. On the recipient's side, equation 3 should be calculated in order to obtain the plain text. It is necessary that $m \geq n$ to ensure that there is only one plain text for each given cipher text [5], [63].

$$y = PK(x) \mid x \in F^n \; and \; y \in F^m \; and \; m \geq n \tag{2}$$

$$x = S^{-1}(F^{-1}(T^{-1}(y))) \tag{3}$$

Additionally, for digital signing usage, the signer should use a secure hash function with the output like $z \in F^n$. Then, the signer should calculate equation 4 to sign the hash output of the message x with the signature s. On the recipient's side, firstly, the hash output of the received message (i.e., x') should be calculated again (shown by z'). Then the recipient should use the sender's public key (i.e., the PK map) and the received signature (shown by s') to obtain the unsigned received hash output (shown by z"). The recipient's calculations are shown by equation 5. Afterward, the recipient can verify validity of the signature by comparing z' with z" [5], [63].

$$z = Hash(x) \; and \; s = T^{-1}(F^{-1}(S^{-1}(z))) \tag{4}$$

$$z' = Hash(x') \; and \; z" = PK(s') \tag{5}$$

One of the main advantages of multivariate cryptography is the speed of execution and simple and efficient implementation. This is because multivariate cryptographic algorithms and digital signatures only require simple arithmetic operations such as addition and multiplication in finite fields. Therefore, developers can effectively implement them on hardware-limited devices. Another advantage of this category of algorithms is the small digital signatures that they produce, which reduces the communication load and makes such algorithms proper for generating digital signatures. On the other hand, as one of the disadvantages of this category of algorithms, we can mention the enormous size of the public key, which reaches hundreds of kilobits in some algorithms of this category, which is larger than keys in many conventional (e.g., RSA) and post-quantum (e.g., lattice-based) asymmetric-key algorithms and creates a significant communication load. Another notable drawback of multivariate cryptographic algorithms is the lack of a proven security level in their case. Although some algorithms in this category have resisted various attacks for years, there is no accurate proof of their security levels [64].

As a few of the most well-known multivariate public-key encryption schemes, we can mention Simple Matrix [65], SRP [66], and EFLASH [67]. Also, as some of the most famous examples of

multivariate digital signature schemes, we can mention Oil-and-Vinegar [64], Unbalanced Oil-and-Vinegar (UOV) [68], FLASH [69], Rainbow [70], and GeMSS [71].

4.5 Isogeny-Based Cryptography

This category is the youngest in post-quantum cryptography, and its mathematical foundation is based on elliptic curve isogeny problems. An "elliptic curve" is a curve in 2-dimensional space that contains all points involved in the equation of $y^2 = x^3 + ax + b$ in a finite field. These points form a finite Abelian group, based on which an algebraic group operation law can be defined, which is called point addition. In an elliptic curve, the result of adding two points is another point on the same elliptic curve. If a particular point is added to itself multiple times, this operation is called "scalar multiplication" and is shown by $k \cdot P$, in that k is an integer, and P is a point on the elliptic curve. Given k and P, it is easy to calculate $Q = k \cdot P$. However, it is a difficult problem to calculate the integer k in the above equation while having P and Q. This problem is known as the Elliptic Curve Discrete Logarithm Problem (ECDLP) and is very time-consuming to be solved by conventional (non-quantum) computers. Therefore, a renowned category of conventional asymmetric-key cryptographic algorithms has been designed based on this problem, which is named Elliptic Curve Cryptography (ECC) [11], [72].

As we explained in Section 3, although solving problems such as discrete logarithms and ECDLP using conventional computers is very time-consuming and has exponential time orders, quantum computers are able to solve these problems in relatively short periods of time (i.e., with polynomial time orders) using Shor's algorithms and some other quantum algorithms similar to them. Therefore, conventional ECC algorithms are not considered secure in the post-quantum era [5], [72].

In efforts to provide quantum-resistant cryptographic algorithms, isogeny-based cryptographic algorithms were presented, which are a relatively new kind of ECC algorithms. An isogeny is a rational map between two or more elliptic curves over a finite field. Two curves are considered isogenous (in other words, isomorphic) if there is an isogeny between them. The degree of an isogeny is its degree as a rational map. Isomorphic curves mutually compose a structure called the isomorphism class and are similar in some characteristics. An isogeny can be considered as a morphism from one isomorphism class to another. Hence, one can construct a graph of all isogenies, where nodes represent the isomorphism classes (composed of isomorphic curves), and edges represent the isogenies between curves. Given two distant nodes on an enormous isogeny graph, it is very difficult to find a path that connects these nodes. It is an example of isogeny-related problems that are used to design algorithms in this category of post-quantum cryptography.

The most effective attack to compute an isogeny between two given isomorphism classes has the time order of $O(n^{1/2})$ on conventional computers and $O(n^{1/3})$ on quantum computers, where n is the degree of the isogeny [72], [73].

Isogeny-based cryptographic algorithms, based on their technical characteristics, generally belong to one of the three major groups of schemes, which are Ordinary Isogeny Diffie–Hellman (OIDH), Supersingular Isogeny Diffie–Hellman (SIDH), and Commutative SIDH (CSIDH). One of the first algorithms in this category, was presented by Rostovtsev and Stolbunov in [74]. This algorithm was founded based on the problem of finding isogenies between ordinary elliptic curves and belongs to the OIDH category. Unfortunately, later, a quantum algorithm was presented in [75], that could solve ordinary isogeny problems with sub-exponential time order and questioned the security of algorithms of this category [5], [17], [72].

In order to design algorithms resistant to the above attack, researchers considered schemes based on isogenies between supersingular elliptic curves, and thus, the SIDH category was born. In 2011, Jao and De-Feo proposed an algorithm for public-key encryption and key encapsulation based on supersingular elliptic curves [76], which is one of the first algorithms in the SIDH category. Later, in 2014, De-Feo et al., in the research [77], improved the algorithm presented in [76] and made it faster to execute and more secure. Another SIDH algorithm is the Supersingular Isogeny Key Encapsulation (SIKE) algorithm [78], [79], which is one of the popular algorithms in post-quantum cryptography. The SIKE consists of two parts: SIKE.PKE for public-key encryption and SIKE.KEM for key encapsulation. Until now, the most effective known attacks against the SIDH problem have exponential time orders. However, some recent research has shown that SIDH keys may be vulnerable to some active attacks and should not be reused [5], [72].

The SIDH problem requires a remarkable mathematical background to be entirely understood. Hence, this category has received less attention from researchers [5]. In 2018, Castryck et al. proposed a new isogeny-based cryptographic algorithm [80], that was designed based on the CSIDH problems and is the first member of CSIDH schemes. According to the structure of this algorithm, it is possible to adapt supersingular isogenies to the OIDH schemes instead of ordinary isogenies. Nevertheless, some recent research shows that the attacks using the quantum algorithm introduced in [75] may still be effective in breaking this algorithm in sub-exponential time [5], [72]. One of the first isogeny-based digital signature algorithms was presented by Dey et al. in 2022 in the article [81]. The security of this algorithm is guaranteed by the hardness of the Commutative Supersingular Isogeny Decisional Diffie–Hellman (CSSIDDH) problem, which is an enhanced version of the CSIDH problem and has been presented by Moriya et al. in the article [82].

Despite some challenges, such as vulnerability to some quantum algorithms and mathematical complexity, researchers are still interested in researching and studying the isogeny-based cryptography area because of some attractive advantages of this category, such as small public key size, small signature size, and low communication cost [8], [72].

4.6 MPC-in-the-Head Digital Signature Schemes

This category is a novel category of post-quantum digital signature schemes, and due to its infancy, most official references, such as [32], have not considered it among the main categories of post-quantum cryptographic algorithms and digital signatures. However, since a remarkable number of the recently presented post-quantum digital signature algorithms belong to this category, we review the characteristics of the algorithms in this category and provide some examples of such algorithms in this section. The functional basis of these algorithms is combining Multi-Party Computation (MPC) techniques and Zero-Knowledge Proof (ZKP) protocols to design a verifiable and unforgeable signature [83], the idea of which was first proposed by Ishai et al. in research [84].

A Zero-Knowledge Protocol (ZKP) is a protocol in which a prover (e.g., P) can convince a verifier (e.g., V) of the correctness of a public (i.e., non-confidential) statement through a public and unsecured channel, without revealing any extra information. For example, user P (the prover) announces to user V (the verifier) that he/she knows a secret value (e.g., x), called the witness, but does not announce x because the channel between users P and V is unsecured and there may be eavesdroppers. Afterward, user V asks questions about x to user P, and user P sends the answers to V. These questions are called random challenges and must be designed so that they do not reveal the secret value (i.e., it is impossible to guess the secret value from these questions and their answers). For example, they can be one-way and irreversible functions, and it is not possible to determine the input value from their output. Finally, after asking questions about the secret value and receiving correct answers, user V is convinced that user P knows the secret value with a very high probability. Zero-knowledge proof protocols are very diverse, and according to the type of questions that the verifier asks the prover and how to ask these questions, there are various kinds and applications of these protocols. One of the most known applications of these protocols is authentication in security systems and cryptocurrency areas [85].

In research [86], Fiat and Shamir presented a method in which, by replacing hash functions instead of the above-mentioned one-way functions, protocols for authentication and digital signature can be created based on ZKP protocols.

Multi-Party Computation (MPC) techniques are protocols in which a user can allow multiple users (i.e., $u_1, u_2, \dots, u_n$), who are mutually distrustful of each other, to communicate with each other and jointly calculate the output of a public (non-confidential) function based on the secret inputs they each have (i.e., $x_1, x_2, \dots, x_n$). In other words, the primary user asks the users $u_1, u_2, \dots, u_n$ to jointly calculate $f(x_1, x_2, \dots, x_n) = y$, while any user must not become aware of another user's input during the communications. A suitable MPC protocol has two features: the first feature is that each user's input value (i.e., x_i's) remains confidential to that user, and the second condition is that the desired output (i.e., y) is correctly calculated [87].

The MPC-in-the-head method, which is the basis for designing digital signatures of this category, is a technique in which a ZKP protocol is implemented based on an MPC protocol. In this method, a prover (e.g., P) who wants to prove its knowledge of a secret value (e.g., x) to a verifier (e.g., V) without revealing x, executes an MPC protocol among n virtual users (i.e., $P_1, P_2, \dots, P_n$). In this operation, the prover P first generates the shares $x_1, x_2, \dots, x_n$ from the original x and gives each x_i to a virtual user such as P_i so that they can calculate the output of a function such as $f(x_1, x_2, \dots, x_n) = y$ together by exchanging information (without revealing the x_i's). Then, P selects a subset of the transcripts exchanged between the P_i's and sends them to V as proof that it really knows the value of x. Then, by checking the correctness and consistency of the received transcripts, V can determine that, with a high probability, P knows the value of x [83].

Some recently introduced digital signature algorithms are based on the MPC-in-the-head method and, therefore, fall into this category. Some of the most known digital signatures in this category are MIRA [88], MiRitH [89], MQOM [90], PERK [91], and RYDE [92]. Large signature size and small public key size are characteristics of the algorithms in this category [83], [93].

4.7 Summary on Post-Quantum Cryptographic Algorithms

In this section, according to what we explained in Sections 4.1 to 4.6, we provide two summary tables about all categories of post-quantum cryptographic algorithms and digital signatures. We designed two separate tables to avoid complexity and make it more convenient for readers. In Table 1, we briefly mention each category's functional basis, advantages, and disadvantages. Then, in Table 2, we introduce some of the most renowned algorithms presented as Key Encapsulation Mechanisms (KEM) and Public-Key Encryption (PKE) algorithms, as well as Digital Signature Schemes (DSSs) in each category.

Table 1. An overview of the algorithm categories in post-quantum cryptography

Category	Functional Basis	Advantages	Disadvantages
Lattice-Based	lattice-related problems	high security level, fast and convenient execution, algorithmic simplicity	vulnerable implementation on conventional computers
Code-Based	error-correcting codes	guaranteed and high security level	huge public key, additional data needed for verifying digital signatures
Hash-Based	hash functions	algorithmic simplicity, fast and convenient execution	huge digital signature, security dependency on utilized hash functions
Multivariate	multivariate polynomial equations	simple and efficient implementation, fast and convenient execution, small digital signature size	huge public key, lack of proven security level
Isogeny-Based	isomorphic elliptic curves	small public key size, small digital signature size, low communication cost	vulnerability to some quantum algorithms, mathematical complexity
MPC-in-the-Head	combination of ZKP and MPC protocols	small public key size	huge digital signature

Table 2. Some renowned examples for each category in post-quantum cryptography

Category	KEM and PKE instances	DSS instances
Lattice-Based	NTRU [41], NTRU-prime [44], SABER [45], CRYSTALS -KYBER [39]	FALCON [46], qTESLA [47], CRYSTALS-DILITHIUM [40]
Code-Based	McEliece [48], BIKE [51], HQC [52], [53], RQC [54]	-
Hash-Based	-	Lamport's OTS [55], Merkle's FTS [57], XMSS [59], SPHINCS [60], SPHINCS+ [62]
Multivariate	Simple Matrix [65], SRP [66], EFLASH [67]	Oil-and-Vinegar [64], Unbalanced Oil-and-Vinegar (UOV) [68], FLASH [69], Rainbow [70], GeMSS [71]
Isogeny-Based	Rostovtsev and Stolbunov's [74], Jao and De-Feo's [76], [77], SIKE [78], [79], Castryck et al.'s algorithm [80]	Dey et al.'s algorithm [81]
MPC-in-the-Head	-	MIRA [88], MiRitH [89], MQOM [90], PERK [91], RYDE [92]

5. Selected Algorithms by NIST

As we mentioned in Section 1, one of the organizations that strives to find and standardize secure and suitable post-quantum cryptographic algorithms and digital signatures is the National Institute of Standards and Technology (NIST) in the United States. In 2016, this organization announced a call for proposed algorithms for asymmetric-key cryptography (i.e., both PKE and KEM) and digital signing, which are resistant to attacks from both conventional and quantum computers, to select the most suitable algorithms among them [94].

In the first round, a total of 82 algorithms were proposed, but only 69 of them met the minimum requirements announced by the NIST, and the rest were discarded. Most of these algorithms were lattice-based or code-based. Through holding technical conferences and receiving comments, the NIST evaluated the proposed algorithms in terms of the criteria it had defined. Some of the most important criteria were the security of algorithms and schemes, computational efficiency, and the amount of hardware required to store and execute each scheme [94], [95].

In 2019, at the end of the first round, the NIST announced a list of 26 algorithms from the initial 69 candidates that were selected to advance to the second round of the competition. In the second round, the NIST evaluated and analyzed candidate algorithms by holding conferences and workshops and receiving comments based on different criteria, the most important of which was security and resistance to various attacks. As in the first round, many of the candidate algorithms were broken against attacks, or their security was reduced, and they were eliminated from the competition [8], [94], [95].

The second round of this competition, which was called the semi-final, ended in 2020, and the NIST announced that the seven selected algorithms, along with the eight alternate algorithms, will move to the third round (i.e., the final) as the finalists. The alternate algorithms were the algorithms that, in the previous round, either showed lower performance than the seven selected algorithms in terms of computation and hardware efficiency or needed more analysis and investigation in terms of security. Also, since five of the seven selected algorithms were from the lattice-based category, the NIST tried to select alternate algorithms from diverse categories. Among the seven algorithms selected in the second round, four were designed for PKE and KEM, and three were intended for digital signature generation [94], [95].

In 2022, the third round of the competition ended, and the NIST announced the list of the four winning algorithms that were selected for standardization and global use in the future years. Three of these algorithms are specific for generating digital signatures, which are CRYSTALS-Dilithium [40], FALCON [46], and SPHINCS+ [62], and one of them is also designed for PKE and KEM, which is called CRYSTALS-KYBER [39]. The Crystal-Kyber and Crystal-Dilithium algorithms

are both lattice-based algorithms that have been selected for their outstanding security and also excellent performance and efficiency, and the NIST expects them to work well in most applications. FALCON is also a lattice-based algorithm that can replace CRYSTALS-Dilithium in cases where smaller signatures are needed. SPHINCS+ is a hash-based digital signature algorithm, and one of the goals of its selection is to avoid dependence only on the category of lattice-based algorithms [5], [8], [17], [94].

In addition to the four algorithms selected at the end of the third round, four other third-round candidates were also selected to compete in the fourth round. The purpose of holding the fourth round is to select at least one post-quantum algorithm for PKE and KEM, which belongs to a category different from lattice-based ones. The four candidate algorithms in the fourth round are: BIKE [51], Classic McEliece [48], HQC [52], [53], and SIKE [78], [79]. BIKE and HQC are both code-based algorithms that can operate as general-purpose PKE and KEM algorithms. The NIST expects to select at most one of these two candidates for standardization at the end of the fourth round. SIKE is an isogeny-based algorithm that is an attractive candidate for standardization due to the small size of the public key and ciphertext. The classic McEliece algorithm, which belongs to the code-based category, was a finalist in the third round. However, because of the enormous size of its public key, which reduces its use case, it was not among the selected algorithms for standardization. Although, this algorithm may be selected at the end of the fourth round for the above purpose. The fourth round has not been finished at the time of writing this article, and the evaluation of candidate algorithms is still ongoing [5], [8], [94]. Table 3 provides an overview of the four rounds of the competition held by the NIST, including statistical information about the candidates in each round [94], [95]. The values related to the third round include the sum of the primary and alternate candidates.

Table 3. An overview of the PQC algorithm selection by the NIST

Category	Round 1		Round 2		Round 3		Roud 4	
	PKE/KEM	DSS	PKE/KEM	DSS	PKE/KEM	DSS	PKE/KEM	DSS
Lattice-Based	21	5	9	3	5	2	0	0
Code-Based	17	2	7	0	3	0	3	0
Hash-Based	0	2	0	1	0	1	0	0
Multivariate	2	8	0	4	0	2	0	0
Isogeny-Based	1	0	1	0	1	0	1	0
MPC-in-the-Head	0	1	0	1	0	1	0	0
Other	4	2	0	1	0	1	0	0
Total	45	19	17	9	9	6	4	0

It is noteworthy that the Rainbow digital signature algorithm [70] from the multivariate category that was one of the primary candidates in the third round (i.e., it had succeeded in passing the first and second rounds) was broken by Ward Beullens on a conventional computer in the research [96]. Additionally, the SIKE public-key encryption algorithm from the isogeny-based category, which was one of the candidates in the fourth round, was broken in 2022 by Castryck and Decru in the research [97], only using a conventional computer. These two breaking means that it is possible that Some algorithms that seem secure in the initial stages of evaluation and analysis may be broken after some time, and their security may be questioned. This fact means that sometimes, several years of research by researchers worldwide is needed to confirm and ensure the correctness and security of a post-quantum cryptographic algorithm.

In addition to the above competition, which was held in three rounds plus an extra fourth round, since 2022, the NIST has started to hold a new competition to find other suitable and secure post-quantum digital signature schemes to diversify its post-quantum signature portfolio [98]. Since among the DSSs selected in the previous competition, two schemes are based on structured lattices, and also according to some potential weaknesses and vulnerabilities of structured lattice-based cryptosystems [99], the NIST has announced its preference for digital signature schemes that belong to the other categories. Additionally, the NIST prefers digital signature schemes with short signatures and fast verification [98].

In the first round of this competition, 50 proposed DSSs were submitted to the NIST, of which 40 met the initial criteria set by the NIST and participated in the competition. Among these 40 DSSs, 7 of them were lattice-based, 6 were code-based, 10 belonged to the multivariate category, one was isogeny-based, 4 were based on symmetric cryptography, 7 belonged to the MPC-in-the-head category, and 5 were based on other problems. The complete list of these DSSs is available in [98].

In 2024, the first round of this competition ended, and among the 40 candidates, 14 of them advanced to the second round. During the first round, some of the candidate DSSs were excluded from the competition due to the efficient attacks that were presented against them. Among the 14 candidates that have entered the second round, one is lattice-based, 2 are code-based, 4 are from the multivariate category, one is isogeny-based, one is based on symmetric cryptography, and 5 belong to the MPC-in-the-head category. The complete list of the second-round candidates is available in [98]. The second round of this competition is still going on when writing this article [98].

6. Implementation Details

In this section, we provide some useful information about implementing and deploying post-quantum cryptographic algorithms and digital signatures. This information is about the four algorithms selected by the NIST after the previously explained three-round competition, which are supposed to be standardized for worldwide and long-term utilization in the future. The information presented in this section is mostly provided by the designers and creators of the algorithms and is available on the NIST's website [94]. The rest has been provided by researchers studying in this field, who have implemented these algorithms and evaluated their performances.

One of the most important specifications of each cryptographic or digital signing algorithm is its strength and resilience against conventional attacks. The metric that the NIST uses to measure the strength and resilience of such algorithms is comparing them with symmetric-key cryptographic algorithms and hash functions in terms of the mentioned specifications. In other words, the NIST expresses the time and computations needed to break cryptographic algorithms based on the time and computations needed to break well-known symmetric-key cryptographic algorithms and hash functions, such as AES and SHA, using conventional attacks, such as brute force and collision attacks, which we described in Section 3. For example, the time and computations required to break the CRYSTALS-Kyber-512 algorithm are nearly equal to what is needed to break the AES-128. Therefore, the NIST has defined five security levels for measuring the security of cryptographic algorithms. With the advent of quantum computing and the realization of the threats from quantum algorithms, the time and computations needed to break criterion symmetric-key algorithms and hash functions reduced, as we described in Section 3. Hence, to use the old metrics in the post-quantum era, it is necessary to adjust them according to the post-quantum situation. More precisely, as we explained in Section 3, by employing Grover's quantum algorithm [19], it is possible to reduce the time order of breaking symmetric-key cryptographic algorithms, which are performed using brute force attacks, from $O(n)$ to $O(n^{1/2})$. Also, using Brassard's quantum algorithms [26], attackers can decrease the time order of performing collision attacks against hash functions from $O(n)$ to $O(n^{1/3})$. Therefore, to update the above security metrics, it is necessary to readjust the quantity of strength and resilience attributed to each criterion symmetric-key algorithm and hash function. After the update, one can use the metrics to show post-quantum security levels of various algorithms. Table 4 demonstrates the five security levels defined by the NIST to compare the post-quantum security levels of different cryptographic algorithms and digital signatures [94], [100].

Table 4. Post-quantum cryptography security levels defined by the NIST

Security level	Benchmark algorithm	Breaking time order (using quantum algorithms)
1	AES-128	$O(2^{64})$
2	SHA-256	$O(2^{85})$
3	AES-192	$O(2^{96})$
4	SHA-384	$O(2^{128})$
5	AES-256	$O(2^{128})$

After familiarity with the definition of security levels, here we provide information about the implementation-related characteristics of the cryptography and digital signing algorithms selected by the NIST. As we explained in Section 5, the NIST selected four post-quantum algorithms through a three-round competition to be standardized for worldwide and long-term use in the future. These four algorithms are CRYSTALS-Kyber [39], CRYSTALS-Dilithium [40], FALCON [46], and SPHINCS+ [62]. CRYSTALS-Kyber is designed for PKE/KEM usage, and the rest are intended for digital signing. For each of these algorithms, their creators have defined different implementation variants and standards. Each of these variants and standards has its own characteristics, such as the public key size, the cipher text block/digital signature size, the count of processor clock cycles for key generation, encryption/signing, and decryption/signature verification. In addition, the levels of security and resistance of the algorithms to common attacks vary depending on the values of the parameters of each of these standards, such as the key size, etc. Table 5 shows the implementation details and security levels of the popular standards for each of the four selected algorithms. The information in this table is mostly provided by the algorithm designers and creators and is available and documented for each algorithm on the NIST website [94]. For the FALCON algorithm, the information about the counts of processor clock cycles is available in the reference [101].

Table 5. Implementation-related details about the NIST's selected PQC algorithms

Algorithm and standard	Pub key size (bits)	Cipher text/ Signature size (bits)	Key-Gen (cycles)	Encryption/ Signing (cycles)	Decryption/ Verifying (cycles)	PQ security Levels
CRYSTALS-Kyber-512	6,400	6,144	655,595	865,256	961,648	1
CRYSTALS-Kyber-768	9,472	8,704	1,087,897	1,373,744	1,491,214	3
CRYSTALS-Kyber-1024	12,544	12,544	1,696,314	2,057,522	2,199,958	5
CRYSTALS-Dilithium-2	10,496	19,360	300,751	1,081,174	327,362	2
CRYSTALS-Dilithium-3	15,616	26,344	544,232	1,713,783	522,267	3
CRYSTALS-Dilithium-5	20,736	36,760	819,475	2,383,399	871,609	5
FALCON-512	7,176	6,016	114,546,135	80,503,242	530,900	1
FALCON-1024	14,344	11,696	365,950,978	165,800,855	1,046,700	5
SPHINCS+-SHA2-128f	256	136,704	5,590,602	138,610,500	7,757,942	1
SPHINCS+-SHA2-128s	256	62,848	358,061,994	2,721,595,944	2,712,044	1
SPHINCS+-SHA2-192f	384	285,312	8,227,944	232,973,880	11,768,382	3
SPHINCS+-SHA2-192s	384	129,792	524,116,024	5,012,149,284	4,333,066	3
SPHINCS+-SHA2-256f	512	398,848	21,763,590	468,188,036	11,934,164	5
SPHINCS+-SHA2-256s	512	238,336	346,844,762	4,499,800,456	6,060,438	5
SPHINCS+-SHAKE-128f	256	136,704	9,649,130	239,793,806	12,909,924	1
SPHINCS+-SHAKE-128s	256	62,848	616,484,336	4,682,570,992	4,764,084	1
SPHINCS+-SHAKE-192f	384	285,312	14,215,518	386,861,992	19,876,926	3
SPHINCS+-SHAKE-192s	384	129,792	898,362,434	8,091,419,556	6,465,506	3
SPHINCS+-SHAKE-256f	512	398,848	36,950,136	763,942,250	19,886,032	5
SPHINCS+-SHAKE-256s	512	238,336	594,081,566	7,085,272,100	10,216,560	5

Also, the creators of the SPHINCS+ algorithm have provided two classes of implementations: the simple implementation and the robust implementation. In order to avoid complexity, only information related to the simple implementations of this algorithm is included in this table. In addition, the ending letter "f" or "s" at the name of the SPHINCS+ standards indicates a fast implementation (i.e., large public key and signature sizes, while low clock cycle counts) or a small implementation (i.e., small public key and signature sizes, while high clock cycle counts). Another detail that is included in the names of the implementation standards for this algorithm is the name of the hash function used by the algorithm [94]. As we said in Section 4.3, this algorithm is a hash-based digital signature algorithm and uses hash functions.

7. Conclusion

Although quantum computing offers numerous benefits and applications, its emergence presents significant challenges and potential threats to some kinds of cryptographic algorithms and digital signatures, raising concerns about their security and reliability. This leads to the inefficiency of many security methods and systems. To address this issue, researchers have designed and proposed novel kinds of cryptographic algorithms, known as post-quantum algorithms, which are resistant to quantum computers and quantum algorithms (i.e., algorithms with special capabilities that specifically execute on quantum computers). In this article, while reviewing different types of conventional (non-quantum) cryptographic algorithms and digital signatures, we explained the reasons leading to a reduction in their security and the threats posed to them by quantum algorithms. Afterward, we described the five major categories of cryptographic algorithms and digital signatures that have been presented so far, along with their functional details and the mathematical basis of each one. We have outlined the advantages and drawbacks of each category and provided examples of the most well-known encryption and digital signature algorithms belonging to each category. Finally, we discussed the competitions organized by the National Organization of Standards and Technology (NIST) in the United States to select suitable and secure algorithms for cryptography and digital signing in the post-quantum world. We have provided useful information about these competitions and the selected algorithms. The primary purpose of producing this article is to provide a comprehensive view of the threats and challenges posed to conventional cryptographic algorithms by quantum computing, as well as the status of post-quantum cryptographic algorithms that have been presented so far. This information is helpful to all individuals and organizations that use cryptography-based security systems and also researchers who are studying in the field of cryptography and cybersecurity. As we emphasized repeatedly throughout the article, post-quantum cryptography still needs a lot of research and development and has numerous deficits and drawbacks that require to be addressed by researchers. Quantum

computers will be commercially available in the near future, although they may already exist in many research centers. Therefore, it is necessary to adopt approaches and take measures as soon as possible to transition from the systems and methods based on conventional cryptography to post-quantum cryptography. Each of the issues discussed in this section can be the subject of further studies and research in the field of cryptography and cybersecurity.

References

1. *Budget of the U.S. Government FISCAL YEAR 2024*. U.S. Office of Management and Budget: [online] Available from: www.whitehouse.gov.

2. Gaithuru, J.N., et al. *A comprehensive literature review of asymmetric key cryptography algorithms for establishment of the existing gap*. in *2015 9th Malaysian Software Engineering Conference (MySEC)*. 2015. IEEE.

3. Boisrond, K., P.M. Tardif, and F. Jaafar, *Ensuring the integrity, confidentiality, and availability of IoT data in Industry 5.0: A Systematic Mapping Study.* IEEE Access, 2024.

4. Joseph, D., et al., *Transitioning organizations to post-quantum cryptography.* Nature, 2022. **605**(7909): p. 237-243.

5. Chamola, V., et al., *Information security in the post quantum era for 5G and beyond networks: Threats to existing cryptography, and post-quantum cryptography.* Computer Communications, 2021. **176**: p. 99-118.

6. Saha, R., et al., *A blockchain framework in post-quantum decentralization.* IEEE Transactions on Services Computing, 2021. **16**(1): p. 1-12.

7. Oliva delMoral, J., et al., *Cybersecurity in Critical Infrastructures: A Post-Quantum Cryptography Perspective.* IEEE Internet of Things Journal, 2024.

8. Allgyer, W., T. White, and T.A. Youssef, *Securing the Future: A Comprehensive Review of Post-Quantum Cryptography and Emerging Algorithms.* SoutheastCon 2024, 2024: p. 1282-1287.

9. Hasija, T., et al. *A survey on nist selected third round candidates for post quantum cryptography*. in *2022 7th International Conference on Communication and Electronics Systems (ICCES)*. 2022. IEEE.

10. Bavdekar, R., et al. *Post quantum cryptography: A review of techniques, challenges and standardizations*. in *2023 International Conference on Information Networking (ICOIN)*. 2023. IEEE.

11. Delfs, H. and H. Knebl, *Introduction to cryptography*. Vol. 2. Springer.

12. Al Busafi, S. and B. Kumar. *Review and analysis of cryptography techniques*. in *2020 9th International Conference System Modeling and Advancement in Research Trends (SMART)*. 2020. IEEE.

13. Tan, T.G., P. Szalachowski, and J. Zhou, *Challenges of post-quantum digital signing in real-world applications: A survey.* International Journal of Information Security, 2022. **21**(4): p. 937-952.

14. Hasija, T., et al. *Symmetric Key Cryptography: Review, Algorithmic Insights, and Challenges in the Era of Quantum Computers*. in *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. 2023. IEEE.

15. Bettale, L., M. De Oliveira, and E. Dottax. *Post-quantum protocols for banking applications*. in *International Conference on Smart Card Research and Advanced Applications*. 2022. Springer.

16. Verchyk, D. and J. Sepúlveda, *A practical study of post-quantum enhanced identity-based encryption.* Microprocessors and Microsystems, 2023. **99**: p. 104828.

17. Gharavi, H., J. Granjal, and E. Monteiro, *Post-quantum blockchain security for the Internet of Things: Survey and research directions.* IEEE Communications Surveys & Tutorials, 2024.

18. Lu, Y. and J. Yang, *Quantum financing system: A survey on quantum algorithms, potential scenarios and open research issues.* Journal of Industrial Information Integration, 2024: p. 100663.

19. Grover, L.K. *A fast quantum mechanical algorithm for database search*. in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. 1996.

20. Zeydan, E., et al. *Recent advances in post-quantum cryptography for networks: A survey*. in *2022 Seventh International Conference On Mobile And Secure Services (MobiSecServ)*. 2022. IEEE.

21. Stiawan, D., et al., *Investigating brute force attack patterns in IoT network.* Journal of Electrical and Computer Engineering, 2019. **2019**(1): p. 4568368.

22. Malina, L., et al., *Post-quantum era privacy protection for intelligent infrastructures.* IEEE Access, 2021. **9**: p. 36038-36077.

23. Fernández-Caramés, T.M., *From pre-quantum to post-quantum IoT security: A survey on quantum-resistant cryptosystems for the Internet of Things.* IEEE Internet of Things Journal, 2019. **7**(7): p. 6457-6480.

24. Sharma, A.K. and S. Mittal. *Cryptography & network security hash function applications, attacks and advances: A review*. in *2019 Third International Conference on Inventive Systems and Control (ICISC)*. 2019. IEEE.

25. Boyer, M., et al., *Tight bounds on quantum searching.* Fortschritte der Physik: Progress of Physics, 1998. **46**(4-5): p. 493-505.

26. Brassard, G., P. Høyer, and A. Tapp. *Quantum cryptanalysis of hash and claw-free functions*. in *LATIN'98: Theoretical Informatics: Third Latin American Symposium Campinas, Brazil, April 20–24, 1998 Proceedings 3*. 1998. Springer.

27. Hamlin, B. and F. Song. *Quantum security of hash functions and property-preservation of iterated hashing*. in *Post-Quantum Cryptography: 10th International Conference, PQCrypto 2019, Chongqing, China, May 8–10, 2019 Revised Selected Papers 10*. 2019. Springer.

28. Shor, P.W. *Algorithms for quantum computation: discrete logarithms and factoring*. in *Proceedings 35th annual symposium on foundations of computer science*. 1994. Ieee.

29. Shor, P.W., *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer.* SIAM review, 1999. **41**(2): p. 303-332.

30. Kampanakis, P. and T. Lepoint. *Vision paper: Do we need to change some things? Open questions posed by the upcoming post-quantum migration to existing standards and deployments*. in *International Conference on Research in Security Standardisation*. 2023. Springer.

31. Hekkala, J., et al., *Implementing post-quantum cryptography for developers.* SN Computer Science, 2023. **4**(4): p. 365.

32. Daniel J. Bernstein, J.B., Erik Dahmen, *Post-Quantum Cryptography*. 2009, Springer Berlin, Heidelberg.

33. Nejatollahi, H., et al., *Post-quantum lattice-based cryptography implementations: A survey.* ACM Computing Surveys (CSUR), 2019. **51**(6): p. 1-41.

34. Canto, A.C., et al., *Error detection schemes assessed on FPGA for multipliers in lattice-based key encapsulation mechanisms in post-quantum cryptography.* IEEE Transactions on Emerging Topics in Computing, 2022. **11**(3): p. 791-797.

35. Peikert, C., *A decade of lattice cryptography.* Foundations and trends® in theoretical computer science, 2016. **10**(4): p. 283-424.

36. Ducas, L., T. Espitau, and E.W. Postlethwaite. *Finding short integer solutions when the modulus is small*. in *Annual International Cryptology Conference*. 2023. Springer.

37. Liu, F., et al., *A survey on lattice-based digital signature.* Cybersecurity, 2024. **7**(1): p. 7.

38. Regev, O., *On lattices, learning with errors, random linear codes, and cryptography.* Journal of the ACM (JACM), 2009. **56**(6): p. 1-40.

39. Bos, J., et al. *CRYSTALS-Kyber: a CCA-secure module-lattice-based KEM*. in *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*. 2018. IEEE.

40. Ducas, L., et al., *Crystals–dilithium: Digital signatures from module lattices.* 2018.

41. Hoffstein, J., *NTRU: A Ring Based Public Key Cryptosystem.* Algorithmic Number Theory (ANTS III), 1998.

42. Hoffstein, J., et al. *NTRUSIGN: Digital signatures using the NTRU lattice*. in *Cryptographers' track at the RSA conference*. 2003. Springer.

43. Kamal, A., et al., *NTRU Algorithm: Nth Degree truncated polynomial ring units*, in *Functional Encryption*. 2021, Springer. p. 103-115.

44. Bernstein, D.J., et al. *NTRU prime: reducing attack surface at low cost*. in *Selected Areas in Cryptography–SAC 2017: 24th International Conference, Ottawa, ON, Canada, August 16-18, 2017, Revised Selected Papers 24*. 2018. Springer.

45. D'Anvers, J.-P., et al. *Saber: Module-LWR based key exchange, CPA-secure encryption and CCA-secure KEM*. in *Progress in Cryptology–AFRICACRYPT 2018: 10th International Conference on Cryptology in Africa, Marrakesh, Morocco, May 7–9, 2018, Proceedings 10*. 2018. Springer.

46. Fouque, P.-A., et al., *Falcon: Fast-Fourier lattice-based compact signatures over NTRU.* Submission to the NIST's post-quantum cryptography standardization process, 2018. **36**(5): p. 1-75.

47. Alkim, E., et al. *The lattice-based digital signature scheme qTESLA*. in *International Conference on Applied Cryptography and Network Security*. 2020. Springer.

48. McEliece, R.J., *A public-key cryptosystem based on algebraic.* Coding Thv, 1978. **4244**: p. 114-116.

49. *Goppa codes.* IEEE Transactions on Information Theory, 2003. **19**(5): p. 590-592.

50. Berlekamp, E., R. McEliece, and H. Van Tilborg, *On the inherent intractability of certain coding problems (corresp.).* IEEE Transactions on Information theory, 1978. **24**(3): p. 384-386.

51. Aragon, N., et al., *BIKE: bit flipping key encapsulation.* 2022.

52. Melchor, C.A., et al., *Hamming quasi-cyclic (HQC).* NIST PQC Round, 2018. **2**(4): p. 13.

53. Deshpande, S., et al. *Fast and efficient hardware implementation of HQC*. in *International Conference on Selected Areas in Cryptography*. 2023. Springer.

54. Bidoux, L., et al., *RQC revisited and more cryptanalysis for rank-based cryptography.* IEEE Transactions on Information Theory, 2023.

55. Lamport, L., *Constructing digital signatures from a one way function.* 1979.

56. Udin, M.N., et al., *Application of Lamport Digital Signature Scheme into the station-to-station protocol.* Malaysian Journal of Computing (MJoC), 2022. **7**(2): p. 1139-1149.

57. Merkle, R.C. *A digital signature based on a conventional encryption function*. in *Conference on the theory and application of cryptographic techniques*. 1987. Springer.

58. Merkle, R.C. *A certified digital signature*. in *Conference on the Theory and Application of Cryptology*. 1989. Springer.

59. Buchmann, J., E. Dahmen, and A. Hülsing. *XMSS-a practical forward secure signature scheme based on minimal security assumptions*. in *Post-Quantum Cryptography: 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29–December 2, 2011. Proceedings 4*. 2011. Springer.

60. Bernstein, D.J., et al. *SPHINCS: practical stateless hash-based signatures*. in *Annual international conference on the theory and applications of cryptographic techniques*. 2015. Springer.

61. Reyzin, L. and N. Reyzin. *Better than BiBa: Short one-time signatures with fast signing and verifying*. in *Australasian Conference on Information Security and Privacy*. 2002. Springer.

62. Bernstein, D.J., et al. *The SPHINCS+ signature framework*. in *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*. 2019.

63. Dey, J. and R. Dutta, *Progress in multivariate cryptography: Systematic review, challenges, and research directions.* ACM Computing Surveys, 2023. **55**(12): p. 1-34.

64. Ding, J. and A. Petzoldt, *Current state of multivariate cryptography.* IEEE Security & Privacy, 2017. **15**(4): p. 28-36.

65. Tao, C., et al. *Simple matrix scheme for encryption*. in *Post-Quantum Cryptography: 5th International Workshop, PQCrypto 2013, Limoges, France, June 4-7, 2013. Proceedings 5*. 2013. Springer.

66. Yasuda, T. and K. Sakurai. *A multivariate encryption scheme with rainbow*. in *Information and Communications Security: 17th International Conference, ICICS 2015, Beijing, China, December 9–11, 2015, Revised Selected Papers 17*. 2016. Springer.

67. Cartor, R. and D. Smith-Tone. *EFLASH: a new multivariate encryption scheme*. in *Selected Areas in Cryptography–SAC 2018: 25th International Conference, Calgary, AB, Canada, August 15–17, 2018, Revised Selected Papers 25*. 2019. Springer.

68. Kipnis, A., J. Patarin, and L. Goubin. *Unbalanced oil and vinegar signature schemes*. in *International Conference on the Theory and Applications of Cryptographic Techniques*. 1999. Springer.

69. Patarin, J., N. Courtois, and L. Goubin. *FLASH, a Fast Multivariate Signature Algorithm: http://www. minrank. org/flash*. in *Topics in Cryptology—CT-RSA 2001: The Cryptographers' Track at RSA Conference 2001 San Francisco, CA, USA, April 8–12, 2001 Proceedings*. 2001. Springer.

70. Ding, J. and D. Schmidt. *Rainbow, a new multivariable polynomial signature scheme*. in *International conference on applied cryptography and network security*. 2005. Springer.

71. Casanova, A., et al., *GeMSS: a great multivariate short signature*. 2017, UPMC-Paris 6 Sorbonne Universités; INRIA Paris Research Centre, MAMBA Team ….

72. Peng, C., et al., *Isogeny-based cryptography: a promising post-quantum technique.* IT Professional, 2019. **21**(6): p. 27-32.

73. Koziel, B., R. Azarderakhsh, and M.M. Kermani, *A high-performance and scalable hardware architecture for isogeny-based cryptography.* IEEE Transactions on Computers, 2018. **67**(11): p. 1594-1609.

74. Rostovtsev, A. and A. Stolbunov, *Public-key cryptosystem based on isogenies.* Cryptology ePrint Archive, 2006.

75. Childs, A., D. Jao, and V. Soukharev, *Constructing elliptic curve isogenies in quantum subexponential time.* Journal of Mathematical Cryptology, 2014. **8**(1): p. 1-29.

76. Jao, D. and L. De Feo. *Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies*. in *Post-Quantum Cryptography: 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29–December 2, 2011. Proceedings 4*. 2011. Springer.

77. De Feo, L., D. Jao, and J. Plût, *Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies.* Journal of Mathematical Cryptology, 2014. **8**(3): p. 209-247.

78. Campagna, M., et al., *Supersingular isogeny key encapsulation*. 2019.

79. Seo, H., et al., *Supersingular isogeny key encapsulation (SIKE) round 2 on ARM Cortex-M4.* IEEE Transactions on Computers, 2020. **70**(10): p. 1705-1718.

80. Castryck, W., et al. *CSIDH: an efficient post-quantum commutative group action*. in *Advances in Cryptology–ASIACRYPT 2018: 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2–6, 2018, Proceedings, Part III 24*. 2018. Springer.

81. Dey, K., et al., *A post-quantum signcryption scheme using isogeny based cryptography.* Journal of Information Security and Applications, 2022. **69**: p. 103280.

82. Moriya, T., H. Onuki, and T. Takagi. *SiGamal: a supersingular isogeny-based PKE and its application to a PRF*. in *Advances in Cryptology–ASIACRYPT 2020: 26th International*

Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7–11, 2020, Proceedings, Part II 26. 2020. Springer.

83. Bettaieb, S., et al., *Enabling PERK and other MPC-in-the-Head Signatures on Resource-Constrained Devices.* IACR Transactions on Cryptographic Hardware and Embedded Systems, 2024. **2024**(4): p. 84-109.

84. Ishai, Y., et al. *Zero-knowledge from secure multiparty computation*. in *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*. 2007.

85. Sun, X., et al., *A survey on zero-knowledge proof in blockchain.* IEEE network, 2021. **35**(4): p. 198-205.

86. Fiat, A. and A. Shamir. *How to prove yourself: Practical solutions to identification and signature problems*. in *Conference on the theory and application of cryptographic techniques*. 1986. Springer.

87. Du, W. and M.J. Atallah. *Secure multi-party computation problems and their applications: a review and open problems*. in *Proceedings of the 2001 workshop on New security paradigms*. 2001.

88. Aragon, N., et al., *MIRA Specifications.* 2023.

89. Adj, G., et al., *MiRitH: Efficient Post-Quantum Signatures from MinRank in the Head.* IACR Transactions on Cryptographic Hardware and Embedded Systems, 2024. **2024**(2): p. 304-328.

90. Benadjila, R., T. Feneuil, and M. Rivain. *MQ on my mind: Post-quantum signatures from the non-structured multivariate quadratic problem*. in *2024 IEEE 9th European Symposium on Security and Privacy (EuroS&P)*. 2024. IEEE.

91. Bettaieb, S., et al., *PERK: compact signature scheme based on a new variant of the permuted kernel problem.* Designs, Codes and Cryptography, 2024: p. 1-27.

92. Aragon, N., et al., *RYDE specifications.* 2023.

93. Bui, D., et al. *Faster Signatures from MPC-in-the-Head*. in *ASIACRYPT 2024-International Conference on the Theory and Application of Cryptology and Information Security*. 2024.

94. *U.S. National Institute of Standards and Technology (NIST), Post-Quantum Cryptography*. Available from: csrc.nist.gov/Projects/post-quantum-cryptography.

95. Moody, D. and A. Robinson, *Cryptographic standards in the post-quantum era.* IEEE Security & Privacy, 2022. **20**(6): p. 66-72.

96. Beullens, W. *Breaking rainbow takes a weekend on a laptop*. in *Annual International Cryptology Conference*. 2022. Springer.

97. Castryck, W. and T. Decru. *An efficient key recovery attack on SIDH*. in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. 2023. Springer.

98. *U.S. National Institute of Standards and Technology (NIST), Post-Quantum Cryptography - Additional Digital Signature Schemes*. Available from: csrc.nist.gov/Projects/pqc-dig-sig.

99. Ravi, P., et al., *Lattice-based key-sharing schemes: A survey.* ACM Computing Surveys (CSUR), 2021. **54**(1): p. 1-39.

100. Saarinen, M.-J.O. *Mobile energy requirements of the upcoming NIST post-quantum cryptography standards*. in *2020 8th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*. 2020. IEEE.

101. Oder, T., et al. *Towards practical microcontroller implementation of the signature scheme Falcon*. in *Post-Quantum Cryptography: 10th International Conference, PQCrypto 2019, Chongqing, China, May 8–10, 2019 Revised Selected Papers 10*. 2019. Springer.