The quantum-resistant digital signature schemes based on new hard problems

Nguyen Kim Tuan[1], Hoang Duc Tho[2*], Nguyen Thi Phuong Hang[3], Pham Van Quoc[4], Luu Hong Dung[3]

[1]Phenikaa School of Computing, Phenikaa University, Duong Noi, Hanoi, Vietnam;
[2]Academy of Cryptography Techniques, 141 Chien Thang, Thanh Tri, Hanoi, Vietnam;
[3]Military Technical Academy, 236 Hoang Quoc Viet, Nghia Do, Hanoi, Vietnam;
[4]VNU University of Science, 334 Nguyen Trai, Thanh Xuan, Hanoi, Vietnam.
[*]Corresponding author: thohd@actvn.edu.vn
Received 12 Mar. 2025; Revised 22 Jul. 2025; Accepted 10 Nov. 2025; Published 28 Nov. 2025
DOI: https://doi.org/10.54939/1859-1043.j.mst.107.2025.114-125

ABSTRACT

In this paper, we propose a family of quantum-resistant digital signature schemes built on novel hard problems defined over finite fields. These problems are, to the best of current knowledge, computationally intractable and therefore not susceptible to Shor's quantum algorithm. Based on these problems, we present three concrete signature schemes with standard key-generation, signing, and verification procedures. We prove the correctness of each scheme and analyze its security against secret-key recovery and forgery attacks. Performance comparisons with representative post-quantum candidates illustrate that the proposed schemes achieve competitive key and signature sizes and efficient signing/verification times, making them attractive for practical deployment.

Keywords: Digital signature; Quantum-resistant; Post-quantum; Discrete logarithm; Novel hard problems.

1. INTRODUCTION

The possibility of large-scale quantum computers has motivated the search for digital signature schemes that remain secure against quantum attacks. This paper follows the approach proposed in [1], constructing signature schemes whose security is based on novel computational problems that, currently, have no known efficient classical or quantum algorithms. By basing signatures on such problems, adversaries cannot exploit Shor's algorithm [2-4] to break the schemes. Building on that idea, we design three practical signature schemes suitable for real-world applications (e.g., alternatives to RSA/DSA) and provide correctness proofs and security analyses. The rest of the paper is organized as follows. Section 2 introduces the discrete logarithm problem and the new hard problems we use. Section 3 describes the three signature schemes and analyzes their correctness and security. Section 4 compares our schemes with existing post-quantum proposals. Section 5 concludes the paper.

2. THE PROPOSED NEW HARD PROBLEMS

2.1. The discrete logarithm problem on the finite field

The Discrete Logarithm Problem (DLP) is described as follows: Given a prime p, a generator g of Z_p^* and an element $y \in Z_p^*$, find the integer x, $0 \le \mathbf{x} \le p - 2$, such that:

$$y = g^x \bmod p$$

2.2. The new hard problems on the finite field

From the DLP on the finite field F_p, we see that if the parameter g is also kept secret, then the discrete logarithm problem on F_p will become an unsolvable problem. In the simplest case, it is possible to choose the secret key x itself for the role of the parameter g, the new hard problem is stated as follows:

The first new hard problem: *Given p is a prime number, for each positive integer y in F_p, find the number x that satisfies the following equation:*

$$y = x^x \bmod p$$

The second new hard problem: *Given p is a prime number, for each pair of positive integers y_1 and y_2 in F_p, find the pair of numbers x_1 and x_2 that satisfy the following equation:*

$$y_1 = (x_1)^{x_2} \bmod p$$

$$y_2 = (x_2)^{x_1} \bmod p$$

The third new hard problem: *Given p is a prime number and (a, b) are positive integers in F_p, find the number x that satisfies the following equation:*

$$(a)^x \equiv (x)^b \bmod p$$

It is easy to see those existing algorithms for the DLP on F_p [5-9] cannot be used to solve this problem. At present, there is no other solution to this problem other than the "brute force attack" method with computational complexity no less than $O(2^n)$, where: n = |p|.

In the proposed digital signature schemes, the first and second new hard problems are used to generate the public and private key pairs in the key generation algorithm, it is also used to generate signatures in the signature generation algorithm, while the third new hard problem is used as the basis for construction the signature verification algorithm.

3. CONSTRUCTING THE QUANTUM-RESISTANT DIGITAL SIGNATURE SCHEMES BASED ON THE NEW HARD PROBLEMS

This section will present the construction of quantum-resistant digital signature schemes based on the new hard problems mentioned in section 2.

3.1. The first scheme

The proposed first scheme here includes the key generation algorithm (algorithm 1.1), the signature generation algorithm (algorithm 1.2) and the signature verification algorithm (algorithm 1.3). These algorithms are presented in the following sections 3.1.1, 3.1.2 and 3.1.3. The proof of the correctness and evaluation of the security of the algorithm are presented in sections 3.1.4 and 3.1.5.

3.1.1. The key generation algorithm

The End–User's public/private key pair is generated by the key generation algorithm based on the domain parameter p, which is a prime number. The domain parameter here can be generated as specified in ISO/IEC 14888-3[10], FIPS 186– 4 [11] or GOST R34.10–94 [12].

To generate a private and public key pair, each signer first needs to choose a secret keys x_1,x_2 : $1 < x_1, x_2 < p–1$ and $GCD(x_2, p–1) = 1$.

The public keys y_1 and y_2 are generated from x_1, x_2 according to the formula:

$$y_1 = (x_1)^{x_2} \bmod p \quad (1)$$

$$y_2 = (x_2)^{x_1} \bmod p \quad (2)$$

The key generation algorithm (algorithm 1.1) of the proposed scheme is described as follows:

Algorithm 1.1:

Input: L_p.

Output: p, x_1, x_2, y_1, y_2.

[1]. **generate** p: len(p) = L_p
[2]. **generate** x_1, x_2: 1< x_1, x_2 < p–1, GCD(x_2, p–1) = 1
[3]. $y_1 \leftarrow (x_1)^{x_2} \bmod p$ **if** (y_1 = 1) **then goto** [2]
[4]. $y_2 \leftarrow (x_2)^{x_1} \bmod p$ **if** (y_2 = 1) **then** goto [2]
[5]. **return** (p, q, x_1, x_2, y_1, y_2)

Note:
– len(.): Function to calculate the length (in bits) of an integer.
– L_p: Length (in bits) of prime number *p*.
– p: System parameter/domain parameter.
– x_1, x_2, y_1, y_2: Private and public keys of the signer.

3.1.2. The signature generation algorithm

Assuming (r, s) is the signature on the message to be signed M. The first component of the signature *r* is calculated according to the following formula:

$$r = \left((x_1)^{x_2+(k)^{-(h-1)}\times(x_1)^{-k}} \times (x_2)^{-h\times(k)^{-h}\times(x_1)^{-(k-1)}} \times (k)^{h\times(k)^{-h}\times(x_1)^{-k}} \right) \bmod p \quad (3)$$

Here: the k is a randomly value in the range (1, p–1) and the h is the representative value (hash value) of the message to be signed M: h = H(M), where: *H()* is the hash function (eg, SHA-1, SHA-256 [13]).

The second component *s* of the signature is calculated according to the formula:

$$s = r \times (k)^h \times (x_1)^k \bmod n \quad (4)$$

Here: $n = p \times (p-1)$.

The signature generation algorithm (algorithm 1.2) of the proposed scheme is described as follows:

Algorithm 1.2:
Input: p, x_1, x_2, M.
Output: (r, s).

[1]. $h \leftarrow H(M)$
[2]. $n \leftarrow p \times (p-1)$
[3]. **generate** k: 1 < k < p–1
[4]. $r \leftarrow \left((x_1)^{x_2+(k)^{-(h-1)}\times(x_1)^{-k}} \times (x_2)^{-h\times(k)^{-h}\times(x_1)^{-(k-1)}} \times (k)^{h\times(k)^{-h}\times(x_1)^{-k}} \right) \bmod p$
[5]. $s \leftarrow r \times (k)^h \times (x_1)^k \bmod n$
[6]. **if** ($(y_1)^s \times (s)^r \bmod p = 1$ **OR** $(y_2)^{r\times h} \times (r)^{s+r} \bmod p = 1$) **then goto** [3]
[7]. **return** (r, s)

Note:
- M: Message to be signed, with: $M \in \{0,1\}^{\infty}$;
- H(.): Hash function, with $H : \{0,1\}^* \mapsto Z_h$, q < h < p.

3.1.3. The signature verification algorithm

The signature verification algorithm of the scheme is construction on the assumption:

$$(y_1)^s \times (s)^r \bmod p = (y_2)^{r\times h} \times (r)^{s+r} \bmod p \quad (5)$$

That is, if M and the signature (r, s) satisfy the equality (5), then the signature is considered

valid, and the message is verified for origin and integrity. Otherwise, the signature is considered forged, and the message to be verified is denied in terms of origin and integrity.

The signature verification algorithm (algorithm 1.3) of the proposed scheme is described as follows:

Algorithm 1.3:

Input: p, y_1, y_2, M, (r, s).

Output: TRUE/FALSE.

[1]. $h \leftarrow H(M)$

[2]. $a \leftarrow (y_1)^s \times (s)^r \bmod p$

[3]. $b \leftarrow (y_2)^{r \times h} \times (r)^{s+r} \bmod p$

[4]. **if** ($a = 1$ **OR** $b = 1$) **then return** (FALSE)

[5]. **if** (a = b) **then return** (TRUE) **else return** (FALSE)

Note:

- M, (r, s): Message and signature to be verified.
- If the result is TRUE, then the integrity and origin of M are asserted. Otherwise, if the result is FALSE, then M is denied for origin and integrity.

3.1.4. The correctness of the proposed signature scheme

What needs to be proved here is:

If:

$$a = (y_1)^s \times (s)^r \bmod p \tag{6}$$

$$b = (y_2)^{r \times h} \times (r)^{s+r} \bmod p \tag{7}$$

Then:

$$a = b$$

Indeed, if the signature and message to be verified are not forged, from (1), (4) and (6), we will have:

$$\begin{aligned} a &= (y_1)^s \times (s)^r \bmod p = (x_1)^{x_2 \times r \times (k)^h \times (x_1)^k} \times \left(r \times (k)^h \times (x_1)^k \right)^r \bmod p \\ &= (x_1)^{x_2 \times r \times (k)^h \times (x_1)^k} \times (k)^{h \times r} \times (x_1)^{k \times r} \times (r)^r \bmod p \end{aligned} \tag{8}$$

From (2), (3), (4) and (7), we get:

$$\begin{aligned} b &= (y_2)^{r \times h} \times (r)^{s+r} \bmod p = (y_2)^{r \times h} \times (r)^s \times (r)^r \bmod p = (x_2)^{x_1 \times r \times h} \times \\ &\times \left((x_1)^{x_2 + (k)^{-(h-1)} \times (x_1)^{-k}} \times (x_2)^{-h \times (k)^{-h} \times (x_1)^{-(k-1)}} \times (k)^{h \times (k)^{-h} \times (x_1)^{-k}} \right)^s \times (r)^r \bmod p \\ &= (x_2)^{x_1 \times r \times h} \times (x_2)^{-x_1 \times h \times (k)^{-h} \times (x_1)^{-k} \times s} \times (x_1)^{x_2 \times s} \times (k)^{h \times (k)^{-h} \times (x_1)^{-k} \times s} \times \\ &\times (x_1)^{k \times (k)^{-h} \times (x_1)^{-k} \times s} \times (r)^r \bmod p = (x_2)^{x_1 \times r \times h} \times (x_2)^{-x_1 \times h \times (k)^{-h} \times (x_1)^{-k} \times r \times (k)^h \times (x_1)^k} \\ &\times (x_1)^{x_2 \times r \times (k)^h \times (x_1)^k} \times (k)^{h \times (k)^{-h} \times (x_2)^{-k} \times r \times (k)^h \times (x_2)^k} \times (x_1)^{k \times (k)^{-h} \times (x_1)^{-k} \times r \times (k)^h \times (x_1)^k} \times \\ &\times (r)^r \bmod p = (x_2)^{x_1 \times r \times h} \times (x_2)^{-x_1 \times r \times h} \times (x_1)^{x_2 \times r \times (k)^h \times (x_1)^k} \times (k)^{h \times r} \times (x_1)^{k \times r} \times \\ &\times (r)^r \bmod p = (x_1)^{x_2 \times r \times (k)^h \times (x_1)^k} \times (x_1)^{k \times r} \times (k)^{h \times r} \times (r)^r \bmod p \end{aligned} \tag{9}$$

From (8) and (9), we have: $a = b$.

3.1.5. The quantum resistance mechanism of the proposed scheme

As mentioned in the introduction section, the type of hard problem used to construct the digital signature scheme here belongs to the class of hard problems with no solution, so for this type of hard problem, Shor's quantum algorithm is ineffective, so quantum attack by Shor's method is not infeasible for the type of scheme proposed here, and that is the quantum resistance mechanism of the proposed type of scheme. The secure of the proposed signature scheme can be further evaluated through its resistance to some types of attacks that will be considered below.

- Secret key attack: In the proposed scheme, attacking the secret key can be performed on the key generation algorithm (algorithm 1.1) and the signature generation algorithm (algorithm 1.2); however, the attacker will encounter the first form of the hard problems mentioned in section 2.2. Therefore, to find the signer's private key from the proposed scheme's key generation and signature generation algorithms, the attacker has no other choice but to solve the above hard problem by the "brute force attack" method.

- Signature forgery attack: From the signature verification algorithm (algorithm 1.3) of the proposed scheme, a set of 2 values (r,s) will be confirmed as a valid signature with the message to be verified M, it satisfies the condition (5).

It can be seen that condition (5) here is the second form of the hard problem mentioned in section 2.2, which is known to be a hard problem (in mathematics) that currently has no other solution than the "brute force" method.

3.2. The second scheme

The proposed second scheme here includes the key generation algorithm (algorithm 2.1), the signature generation algorithm (algorithm 2.2) and the signature verification algorithm (algorithm 2.3). These algorithms are presented in the following sections 3.2.1, 3.2.2 and 3.2.3. The proof of the correctness and evaluation of the security of the algorithm are presented in sections 3.2.4 and 3.2.5.

3.2.1. The key generation algorithm

In this scheme, the End–User's public/private key pair is generated by the key generation algorithm based on the set of domain parameters, including a pair of prime numbers p, q satisfy: q|(p - 1). The domain parameters here can be generated as specified in ISO/IEC 14888-3[10], FIPS 186–4 [11] or GOST R34.10–94 [12]. Similar to the DSA signature scheme [11], the use of the subgroup Z_q here is intended to reduce the magnitude of the exponent in the power operations, thereby allowing for increasing the efficiency of the algorithm. In addition, it also allows for reducing the size of the signature generated by this scheme.

To generate a private and public key pair, each signer first needs to choose $\alpha_1, \alpha_2 \in Z_p^*$, then compute the secret keys x_1, x_2 according to the formula:

$$x_1 = (\alpha_1)^{\frac{p-1}{q}} \bmod p$$

$$x_2 = (\alpha_2)^{\frac{p-1}{q}} \bmod p$$

The public keys y_1 and y_2 are generated from x_1, x_2 and p, q according to the formulas:

$$y_1 = (x_1)^{x_2} \bmod p \quad (10)$$

$$y_2 = (x_2)^{-x_1} \bmod p \quad (11)$$

The key generation algorithm (algorithm 2.1) of the proposed scheme is described as follows:

Algorithm 2.1:

Input: L_p, L_q.
Output: p, q, x_1, x_2, y_1, y_2.

[1]. generate p, q: len(p)= L_p, len(q) = L_q, q|(p–1)
[2]. select α_1: 1< α_1 <p
[3]. $x_1 \leftarrow (\alpha_1)^{\frac{p-1}{q}} \bmod p$ if (x_1 = 1) then goto [2]
[4]. select α_2: 1 < α_2 <p
[5]. $x_2 \leftarrow (\alpha_2)^{\frac{p-1}{q}} \bmod p$ if (x_2 = 1) then goto [4]
[6]. $y_1 \leftarrow (x_1)^{x_2} \bmod p$ if (y_1 = 1) then goto [2]
[7]. $y_2 \leftarrow (x_2)^{-x_1} \bmod p$ if (y_2 = 1) then goto [2]
[8]. return (p, q, x_1, x_2, y_1, y_2)

Note:
- len(.): Function to calculate the length (in bits) of an integer;
- L_p, L_q: Length (in bits) of prime numbers p and q;
- p, q: System parameter/domain parameters;
- x_1, x_2, y_1, y_2: Private and public keys of the signer.

3.2.2. The signature generation algorithm

Assuming (r, s) is the signature on the message to be signed M. The first component of the signature *r* is calculated according to the following formula:

$$r = \left((x_1)^{x_2} \times (x_2)^{(x_1 \times h + k) \times (k)^{-h} \times (x_2)^{-k}} \times (k)^{h \times (k)^{-h} \times (x_2)^{-k}} \right) \bmod n \quad (12)$$

Here: k is a randomly chosen value in the range (1,q) and h is the representative value (hash value) of the message to be signed M: h = H(M).

The second component *s* of the signature is calculated according to the following formula:

$$s = r \times (k)^h \times (x_2)^k \bmod n \quad (13)$$

Here: $n = p \times q$.

The signature generation algorithm (algorithm 2.2) of the proposed scheme is described as follows:

Algorithm 2.2:

Input: p, q, x_1, x_2, M.
Output:(r, s).

[1]. $h \leftarrow H(M)$
[2]. $n \leftarrow p \times q$
[3]. **generate** k: 1 < k < q
[4]. $r \leftarrow \left((x_1)^{x_2} \times (x_2)^{(x_1 \times h + k) \times (k)^{-h} \times (x_2)^{-k}} \times (k)^{h \times (k)^{-h} \times (x_2)^{-k}} \right) \bmod n$
[5]. $s \leftarrow r \times (k)^h \times (x_2)^k \bmod n$
[6]. **if** $((y_1)^s \times (s)^r \bmod p = 1$ **OR** $(y_2)^{r \times h} \times (r)^{s+r} \bmod p = 1)$ **then goto** [3]
[7]. **return** (r,s)

Note:
- M: Message to be signed, with: $M \in \{0,1\}^{\infty}$.
- H(.): Hash function, with $H : \{0,1\}^* \mapsto Z_h$, q < h < p.

3.2.3. The signature verification algorithm

The signature verification algorithm of the scheme is construction on the assumption:

$$(y_1)^s \times (s)^r \bmod p = (y_2)^{r\times h} \times (r)^{s+r} \bmod p \tag{14}$$

That is, if M and the signature (r, s) satisfy the equality (14), then the signature is considered valid, and the message is verified for origin and integrity. Otherwise, the signature is considered forged, and the message to be verified is denied in terms of origin and integrity.

The signature verification algorithm (algorithm 2.3) is described as follows:

Algorithm 2.3

Input: p, q, y_1, y_2, M, (r, s).
Output: TRUE/FALSE.

[1]. $h \leftarrow H(M)$

[2]. $a \leftarrow (y_1)^s \times (s)^r \bmod p$

[3]. $b \leftarrow (y_2)^{r\times h} \times (r)^{s+r} \bmod p$

[4]. **if** ($a = 1$ **OR** $b = 1$) **then return** (FALSE)

[5]. **if** (a = b) **then return** (TRUE) **else return** (FALSE)

Note: If the result is TRUE, then the integrity and origin of M are asserted. Otherwise, if the result is FALSE, then M is denied for origin and integrity.

3.2.4. The correctness of the proposed signature scheme

What needs to be proved here is:

If:
$$a = (y_1)^s \times (s)^r \bmod p \tag{15}$$
$$b = (y_2)^{r\times h} \times (r)^{s+r} \bmod p \tag{16}$$

Then: a = b

Indeed, if the signature and message to be verified are not forged, from (10), (13) and (15) we will have:

$$\begin{aligned} a &= (y_1)^s \times (s)^r \bmod p = (x_1)^{x_2\times r\times (k)^h\times (x_2)^k} \times \left(r\times (k)^h \times (x_2)^k\right)^r \bmod p \\ &= (x_1)^{r\times (k)^h\times (x_2)^{k+1}} \times (k)^{h\times r} \times (x_2)^{k\times r} \times (r)^r \bmod p \end{aligned} \tag{17}$$

From (11), (12), (13) and (16), we get:

$$\begin{aligned} b &= (y_2)^{r\times h} \times (r)^{s+r} \bmod p = (y_2)^{r\times h} \times (r)^s \times (r)^r \bmod p \\ &= (x_2)^{-x_1\times r\times h} \times \left((x_1)^{x_2} \times (x_2)^{(x_1\times h+k)\times (k)^{-h}\times (x_2)^{-k}} \times (k)^{h\times (k)^{-h}\times (x_2)^{-k}}\right)^s \times (r)^r \bmod p \\ &= (x_2)^{-x_1\times r\times h} \times (x_2)^{x_1\times h\times (k)^{-h}\times (x_2)^{-k}\times s} \times (x_1)^{x_2\times s} \times (k)^{h\times (k)^{-h}\times (x_2)^{-k}\times s} \times (x_2)^{k\times (k)^{-h}\times (x_2)^{-k}\times s} \\ &\times (r)^r \bmod p \\ &= (x_2)^{-x_1\times r\times h} \times (x_2)^{x_1\times h\times (k)^{-h}\times (x_2)^{-k}\times r\times (k)^h\times (x_2)^k} \times (x_1)^{x_2\times r\times (k)^h\times (x_2)^k} \\ &\times (k)^{h\times (k)^{-h}\times (x_2)^{-k}\times r\times (k)^h\times (x_2)^k} \times (x_2)^{k\times (k)^{-h}\times (x_2)^{-k}\times r\times (k)^h\times (x_2)^k} \times (r)^r \bmod p \\ &= (x_2)^{-x_1\times r\times h} \times (x_2)^{x_1\times r\times h} \times (x_1)^{r\times (k)^h\times (x_2)^{k+1}} \times (k)^{h\times r} \times (x_2)^{k\times r} \times (r)^r \bmod p \\ &= (x_1)^{r\times (k)^h\times (x_2)^{k+1}} \times (x_2)^{k\times r} \times (k)^{h\times r} \times (r)^r \bmod p \end{aligned} \tag{18}$$

From (17) and (18), we have: $a = b$.

Thus, the correctness of the scheme has been proved.

3.2.5. The quantum resistance mechanism of the proposed scheme

The analysis and evaluation of the quantum resistant mechanism as well as the secure of this scheme can be performed similarly to the first scheme mentioned in section 3.1.5.

3.3. The third scheme

The proposed scheme here includes the key generation algorithm (algorithm 3.1), the signature generation algorithm (algorithm 3.2) and the signature verification algorithm (algorithm 3.3). These algorithms are presented in the following sections 3.3.1, 3.3.2 and 3.3.3. The proof of the correctness and evaluation of the security of the algorithm are presented in sections 3.3.4 and 3.3.5.

3.3.1. The key generation algorithm

The End-User's public/private key pair is generated by the key generation algorithm based on the set of domain parameters, including a pair of prime numbers p, q satisfy: q|(p - 1). The domain parameters here can be generated as specified in ISO/IEC 14888-3[10], FIPS 186 – 4 [11] or GOST R34.10 – 94 [12].

To generate a private and public key pair, each signer first needs to choose a value $\alpha \in Z_p^*$, then compute the secret key *x* according to the formula:

$$x = (\alpha)^{\frac{p-1}{q}} \bmod p$$

The public key y is generated from x and p, q according to the formula:

$$y = (x)^{(x)^{-1}} \bmod p \tag{19}$$

The key generation algorithm (algorithm 3.1) of the proposed scheme is described as follows:

Algorithm 3.1:

Input: L_p, L_q.
Output: p, q, x, y.

[1]. generate p, q: len(p) = L_p, len(q) = L_q, q|(p–1)
[2]. select α: 1 < α < p
[3]. $x \leftarrow (\alpha)^{\frac{p-1}{q}} \bmod p$ if (x = 1) then goto [2]
[4]. $y \leftarrow (x)^{(x)^{-1}} \bmod p$ if (y = 1) then goto [2]
[5]. return (p, q, x, y)

Note:
- len(.): Function to calculate the length (in bits) of an integer.
- L_p, L_q: Length (in bits) of prime numbers *p* and *q*.
- p, q: System parameter/domain parameter.
- x, y: Private and public key of the signer.

3.3.2. The signature generation algorithm

Assuming (r,s) is the signature on the message to be signed M. The first component of the signature *r* is calculated according to the following formula:

$$r = (x)^{x^{-1} \times \left(h + k^{-(x-1)} \times x^{-(k-1)}\right)} \times (k)^{k^{-x} \times x^{-(k-1)}} \bmod n \tag{20}$$

Here, k is a randomly chosen value in the range (1, q) and h is the representative value (hash value) of the message to be signed M: h = H(M).

The second component s of the signature *s* is calculated according to the following formula:

$$s = r \times (k)^x \times (x)^k \bmod n \tag{21}$$

Here: $n = p \times q$.

The signature generation algorithm (algorithm 3.2) of the proposed scheme is described as follows:

Algorithm 3.2:

Input: p, q, x, M.
Output:(r, s).

[1]. $h \leftarrow H(M)$

[2]. $n \leftarrow p \times q$

[3]. **select** β: 1 < β < p

[4]. $k \leftarrow (\beta)^{\frac{p-1}{q}} \bmod p$ if (k = 1) then goto [3]

[5]. $r \leftarrow (x)^{x^{-1} \times \left(h + k^{-(x-1)} \times x^{-(k-1)}\right)} \times (k)^{k^{-x} \times x^{-(k-1)}} \bmod n$

[6]. $s \leftarrow r \times (k)^x \times (x)^k \bmod n$

[7]. **if** $((y)^{s \times h} \times (s)^r \bmod p = 1$ **OR** $(r)^{s+r} \bmod p = 1)$ **then goto** [3]

[8]. **return** (r,s)

Note:

- M: Message to be signed, with: $M \in \{0,1\}^{\infty}$.

- H(.): hash function, with $H : \{0,1\}^* \mapsto Z_h$, q < h < p .

3.3.3. The signature verification algorithm

The signature verification algorithm of the scheme is construction on the assumption:

$$(y)^{s \times h} \times (s)^r \bmod p = (r)^{s+r} \bmod p \tag{22}$$

That is, if M and the signature (r, s) satisfy the equality (22), then the signature is considered valid, and the message is verified for origin and integrity. Otherwise, the signature is considered forged, and the message to be verified is denied in terms of origin and integrity.

The signature verification algorithm (algorithm 3.3) of the proposed scheme is described as follows:

Algorithm 3.3:

Input: p, q, y, M, (r, s).
Output: TRUE/FALSE.

[1]. $h \leftarrow H(M)$

[2]. $a \leftarrow (y)^{s \times h} \times (s)^r \bmod p$

[3]. $b \leftarrow (r)^{s+r} \bmod p$

[4]. **if** ($a = 1$ **OR** $b = 1$) **then return** (FALSE)

[5]. **if** (a = b) **then return** (TRUE)} **else return** (FALSE)

Note:

- M,(r,s): Message and signature to be verified.

- If the result is TRUE, then the integrity and origin of M are asserted. Otherwise, if the result is FALSE, then M is denied for origin and integrity.

3.3.4. The correctness of the proposed scheme

What needs to be proved here is:

If:

$$a = (y)^{s\times h} \times (s)^{r} \bmod p \tag{23}$$

$$b = (r)^{s+r} \bmod p \tag{24}$$

Then:

$$a = b$$

Indeed, if the signature and message to be verified are not forged, from (19), (21) and (24) we will have:

$$\begin{aligned} a &= (y)^{s\times h} \times (s)^{r} \bmod p = (x)^{(x)^{-1}\times r\times (k)^{x}\times (x)^{k}\times h} \times \left(r\times (k)^{x} \times (x)^{k}\right)^{r} \bmod p \\ &= (x)^{(x)^{-1}\times r\times (k)^{x}\times (x)^{k}\times h} \times (k)^{r\times x} \times (x)^{k\times r} \times (r)^{r} \bmod p \end{aligned} \tag{25}$$

From (20), (21) and (25), we get:

$$\begin{aligned} b &= (r)^{s+r} \bmod p = (r)^{s} \times (r)^{r} \bmod p \\ &= (r)^{r\times (k)^{x}\times (x)^{k}} \times (r)^{r} \bmod p \\ &= \left((x)^{(x)^{-1}\times \left(h+(k)^{-(x-1)}\times (x)^{-(k-1)}\right)} \times (k)^{(k)^{-x}\times (x)^{-k-1}} \right)^{r\times (k)^{x}\times (x)^{k}} \times (r)^{r} \bmod p \\ &= \left((x)^{(x)^{-1}\times h} \times (x)^{k\times (k)^{-x}\times (x)^{-k}} \times (k)^{x\times (k)^{-x}\times (x)^{-k}} \right)^{r\times (k)^{x}\times (x)^{k}} \times (r)^{r} \bmod p \\ &= (x)^{(x)^{-1}\times r\times (k)^{x}\times (x)^{k}\times h} \times (k)^{r\times x} \times (x)^{k\times r} \times (r)^{r} \bmod p \end{aligned} \tag{26}$$

From (25) and (26) we have: $a = b$.

Thus, the correctness of the scheme has been proved.

3.3.5. The quantum resistance mechanism of the proposed scheme

The analysis and evaluation of the quantum-resistant mechanism as well as the secure of this scheme can be performed similarly to the first scheme mentioned in section 3.1.5.

3.4. Comparison with existing post-quantum signature schemes

Clearly, grid search ensures no combination in the search grid is missed, but it becomes computationally expensive in high-dimensional spaces. Random search efficiently eliminates low-potential regions by sampling randomly, while Bayesian optimization fine-tunes intelligently by balancing exploration and exploitation using surrogate models, as shown in NIPS papers. The combination of these three stages allows for comprehensive coverage of the search space, reduces the number of evaluations, and finds optimal hyperparameters within a limited computational budget. This makes it a robust, efficient, and scalable optimization process applicable to various machine learning models and tasks.

Table 1 shows that the proposed post-quantum digital signature scheme delivers competitive performance compared to current standard schemes. In terms of size, it requires only a 256-bit public key and a signature of about 512 bits, significantly smaller than Dilithium2 (1.3 KB/2.4 KB) or SPHINCS+ (48 bytes/7.9 KB), thus greatly reducing the burden on resource-constrained devices and networks. Regarding speed, the average signing and verification times range from 0.5–1.2 ms and 0.4–0.9 ms, respectively, which are comparable to or faster than Falcon and

SPHINCS+, and approach Dilithium2, one of the most optimized schemes available today. Unlike lattice-based schemes, our design does not rely on matrix operations, which simplifies implementation and conserves resources.

Table 1. *Comparison with existing post-quantum signature schemes.*

Criteria	Proposed scheme	CRYSTALS-Dilithium2 [18]	Falcon-512 [19]	SPHINCS+-128s [20]
Key size (public/private)	256 bit/ 256 bit	1312/ 2528 byte	897/ 1281 byte	32/ 64 byte
Signature size	~ 512 bit	2420 byte	666 byte	7856 byte
Signing time (ms)	0.5 – 1.2	0.65	2.2	12.5
Verification time (ms)	0.4 – 0.9	0.16	1.5	12.4
Mathematical basis	Proposed hard problem	Lattice	Lattice	Hash-based

In summary, the proposed scheme achieves a strong balance between post-quantum security, performance, and deployability, making it a highly promising candidate for practical applications.

4. CONCLUSIONS

We have presented a family of digital signature schemes whose security is based on novel hard problems over finite fields. Under the assumption that these problems are intractable for both classical and quantum algorithms, the proposed schemes resist known quantum attacks such as those based on Shor's algorithm. We provided three concrete constructions, proved correctness, and discussed their resistance to key-recovery and forgery attacks. Experimental performance data indicate that the schemes can achieve compact key/signature sizes and efficient signing/verification times. Future work includes: (i) A formal hardness analysis of the new problems; (ii) A detailed security reduction where possible; and (iii) Implementation studies and side-channel resistance evaluation.

REFERENCES

[1]. Luu Hong Dung, Nguyen Kim Tuan, Nong Phuong Trang, Pham Van Quoc, *"A solution for constructing quantum-resistant digital signature schems"*, Journal of Military Science and Technology, ISSN 1859–1043, pp. 108–118, (2024), DOI: 10.54939/1859-1043.j.mst.CSCE8.2024.108-118.

[2]. P. W. Shor, *"Algorithms for quantum computation: Discrete logarithms and factoring"*, Proceedings of the 35th Symposium on Foundations of Computer Science, pp. 124–134, (1994).

[3]. P. W. Shor, *"Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer"*, SIAM Journal of Computing, vol. 26, no. 5, pp. 1484–1509, (1997).

[4]. M. Eker, *"Modifying Shor's algorithm to compute short discrete logarithms"*, IACR ePrint Archive, Report 2016/1128, (2016).

[5]. L. C. Washington, *"Elliptic Curves. Number Theory and Cryptography"*, Chapman & Hall/CRC, (2008).

[6]. Jeffrey Hoffstein, Jill Pipher, Joseph H. Silverman, *"An Introduction to Mathematical Cryptography"*, Springer-Verlag, ISBN 978-0-387-77993-5, (2008).

[7]. J. Talbot, D. Welsh, *"Complexity and Cryptography: An Introduction"*, Cambridge University Press, (2006).

[8]. I. Shparlinski, *"Cryptographic Applications of Analytic Number Theory. Complexity Lower Bounds and Pseudorandomness"*, Birkhäuser, (2003).

[9]. S. S. Wagstaff, *"Cryptanalysis of Number Theoretic Ciphers"*, Chapman & Hall/CRC, (2003).

[10]. ISO/IEC 14888-3, *"Information technology – Security techniques – Digital signatures with appendix"*, (2006).

[11]. National Institute of Standards and Technology, *"NIST FIPS PUB 186-4. Digital Signature Standard"*, U.S. Department of Commerce, (2013).

[12]. GOST R 34.10-94, *"Russian Federation Standard. Information Technology. Cryptographic Data*

Security", Government Committee for Standards, (1994).
[13]. Federal Information Processing Standards Publication 180-4 (FIPS PUB 180-4), *"Secure Hash Standard (SHS)"*, (2015).
[14]. ISO/IEC 15946, *"Information technology – Security techniques – Cryptographic Techniques Based on Elliptic Curves"*, (1999).
[15]. ANSI X9.62, *"Public Key Cryptography for the Financial Services Industry: Elliptic Curve Digital Signature Algorithm (ECDSA)"*, (1999).
[16]. National Institute of Standards and Technology, *"NIST FIPS PUB 186-4. Digital Signature Standard"*, (2013).
[17]. GOST R34.10–2012, *"Russian Federation Standard Information Technology"*, Government Committee for Standards, (2012).
[18]. L. Ducas et al., *"CRYSTALS-Dilithium: A Lattice-Based Digital Signature Scheme"*, NIST PQC Round 3 Submission, (2020), https://pq-crystals.org/dilithium/.
[19]. P. A. Fouque et al., *"Falcon: Fast-Fourier Lattice-Based Compact Signatures Over NTRU"*, NIST PQC Round 3 Submission, (2020), https://falcon-sign.info/.
[20]. A. Hülsing et al., *"SPHINCS+: Submission to the NIST Post-Quantum Project"*, (2020), https://sphincs.org/.

TÓM TẮT

Lược đồ chữ ký kháng lượng tử xây dựng trên các bài toán khó mới

Trong bài báo này, các tác giả đề xuất các lược đồ chữ ký kháng lượng tử xây trên một số bài toán khó mới, thuộc nhóm bài toán khó mà hiện tại không có cách giải. Do đó, các thuật toán được xây dựng theo giải pháp đề xuất ở đây có thể chống lại các cuộc tấn công lượng tử dựa trên thuật toán do P. Shor đề xuất. Ngoài khả năng kháng lượng tử, các lược đồ chữ ký được đề xuất ở đây còn có thể sử dụng như các lược đồ chữ ký số đang được sử dụng rộng rãi trong các ứng dụng thực tế hiện nay (RSA, DSA,...).

Từ khóa: Chữ ký số; Kháng lượng tử; Hậu lượn tử; Logarit rời rạc; Bài toán khó mới.