
HOMOMORPHIC POLYNOMIAL PUBLIC KEY CRYPTOGRAPHY FOR QUANTUM-SECURE DIGITAL SIGNATURE

A PREPRINT

 **Randy Kuang***

Quantropi Inc.
Ottawa, ON, Canada
randy.kuang@quantropi.com

Maria Perepechaenko

Quantropi Inc.
Ottawa, ON, Canada
maria.perepechaenko@quantropi.com

Mahmoud Sayed

Systems and Computer Engineering
Carleton University
Ottawa, ON, Canada
mahmoudsayed3@cmail.carleton.ca

Dafu Lou

Quantropi Inc.
Ottawa, ON, Canada
dafu.lou@quantropi.com

November 17, 2023

ABSTRACT

In their 2022 study, Kuang et al. introduced the Multivariable Polynomial Public Key (MPPK) cryptography, a quantum-safe public key cryptosystem leveraging the mutual inversion relationship between multiplication and division. MPPK employs multiplication for key pair construction and division for decryption, generating public multivariate polynomials. Kuang and Perepechaenko expanded the cryptosystem into the Homomorphic Polynomial Public Key (HPPK), transforming product polynomials over large hidden rings using homomorphic encryption through modular multiplications. Initially designed for key encapsulation mechanism (KEM), HPPK ensures security through homomorphic encryption of public polynomials over concealed rings. This paper extends its application to a digital signature scheme. The framework of HPPK KEM can not be directly applied to the digital signatures due to the different nature of verification procedure compared to decryption procedure. Thus, in order to use the core ideas of the HPPK KEM scheme in the framework of digital signatures, the authors introduce an extension of the Barrett reduction algorithm. This extension transforms modular multiplications over hidden rings into divisions in the verification equation, conducted over a prime field. The extended algorithm non-linearly embeds the signature into public polynomial coefficients, employing the floor function of big integer divisions. This innovative approach overcomes vulnerabilities associated with linear relationships of earlier MPPK DS schemes. The security analysis reveals exponential complexity for both private key recovery and forged signature attacks, taking into account that the bit length of the rings is twice that of the prime field size. The effectiveness of the proposed Homomorphic Polynomial Public Key Digital Signature (HPPK DS) scheme is illustrated through a practical toy example, showcasing its intricate functionality and enhanced security features.

Keywords Post-Quantum Cryptography · PQC · Quantum Cryptography · KEM · Digital Signature · Barrett Reduction Algorithm

1 Introduction

Kuang in 2021 [1] proposed a public key scheme called deterministic polynomial public key or DPPK for key exchange. DPPK essentially consists of two solvable univariate polynomials $f(x)$ and $h(x)$ multiplying with a randomly chosen

* 1545 Carling Av, Suite 620, Ottawa, ON K1Z 8P9, Canada

polynomial $B(x)$. The public key is constructed through polynomial multiplications over a finite field \mathbb{F}_p excluding their constant terms. It is secure against the key recovery from the public key but it is insecure against the secret recovery with the deterministic factoring technique reported by Evdokimov in 1994 using Generalized Riemann Hypothesis or GRH [2]. Later, Kuang and Barbeau [3, 4] proposed to change the univariate polynomial $B(x)$ to a multivariate polynomial $B(x, u_1, \dots, u_m)$ with variable x for the secret and u_1, \dots, u_m for random noise values to produce two multivariate polynomials $P(x, u_1, \dots, u_m)$ and $Q(x, u_1, \dots, u_m)$ by excluding both constant terms and highest order terms. This updated variant is called the multivariate polynomial public key or MPPK for key encapsulation mechanism or KEM. In MPPK KEM, the excluded terms in polynomials $P(x, u_1, \dots, u_m)$ and $Q(x, u_1, \dots, u_m)$ form separate public key elements called noise functions, namely $N_0(u_1, \dots, u_m)$ associated with the constant terms and $N_n(x, u_1, \dots, u_m)$ associated with the highest order terms. Both $N_0(u_1, \dots, u_m)$ and $N_n(x, u_1, \dots, u_m)$ are encrypted with randomly chosen private key elements R_0 and $R_n \in \mathbb{F}_p$. By leveraging the noise functions together with all public key elements, the Gaussian elimination technique can be used to eliminate unknown coefficients from $B(x, u_1, \dots, u_m)$ and create equation system of unknown coefficients from univariate polynomials $f(x)$ and $h(x)$, potentially reducing the private key security.

Kuang, Perepechaenko, and Barbeau in 2022 [5] proposed to encrypt the coefficients of the noise functions $N_0(u_1, \dots, u_m)$ and $N_n(x, u_1, \dots, u_m)$ over a hidden ring $\mathbb{Z}/S\mathbb{Z}$ with self-shared secret key values R_0 and R_n respectively. The ring size is required to be at least $2\times$ bigger than the size of the field prime p . Kuang and Perepechaenko in 2023 [6] further extended this MPPK KEM by encrypting the entire public key elements over a hidden ring and renamed it as Homomorphic Polynomial Public Key or HPPK KEM because the encryption over the hidden ring holds the homomorphic property. Kuang and Perepechaenko in 2023 [7] proposed a further variant of the HPPK KEM over dual hidden rings with each public polynomials $P(x, u_1, \dots, u_m)$ and $Q(x, u_1, \dots, u_m)$ encrypted over their own separate rings. The homomorphic symmetric encryption essentially turns public key cryptography into symmetric encryption with the self-shared keys.

On the other hand, Kuang, Perpechaenko, and Barbeau in 2022 [8] proposed their digital signature scheme or MPPK/DS based on the identity equation $f(x)Q(x, u_1, \dots, u_m) = h(x)P(x, u_1, \dots, u_m) \pmod p$, with $P(x, u_1, \dots, u_m) = f(x)B(x, u_1, \dots, u_m)$ and $Q(x, u_1, \dots, u_m) = h(x)B(x, u_1, \dots, u_m)$ over the prime field \mathbb{F}_p . Later, Kuang and Perpechaenko optimized MPPK/DS for parameter selections [9]. The identity equation in MPPK/DS offers a way to establish the signature verification but it leaves a way for the forged signature reported later by Guo in 2023 [10].

2 Contribution

In this paper, we propose a novel digital signature scheme based on the HPPK KEM using the symmetric encryption over dual hidden rings with secret keys R_1 from $\mathbb{Z}/S_1\mathbb{Z}$ and R_2 from $\mathbb{Z}/S_2\mathbb{Z}$ and considering

$$R_1^{-1}R_1 = 1 \pmod{S_1}, R_2^{-1}R_2 = 1 \pmod{S_2} \quad (1)$$

with condition $\gcd(R_1, S_1) = 1$ and $\gcd(R_2, S_2) = 1$. We can then insert Eq. (1) into the identity equation $f(x)Q(x, u_1, \dots, u_m) = h(x)P(x, u_1, \dots, u_m) \pmod p$ and turn it into a conditional identity equation with knowing the secret keys $R_1 \in \mathbb{Z}/S_1\mathbb{Z}$ and $R_2 \in \mathbb{Z}/S_2\mathbb{Z}$. The secret keys R_1, S_1 and R_2, S_2 are used to encrypt coefficients of the product polynomials $P(x, u_1, \dots, u_m)$ and $Q(x, u_1, \dots, u_m)$, respectively, using modular multiplicative operations and the produced ciphertext coefficients are used to derive public key polynomials. The moduli S_1 and S_2 representing the hidden rings can't be shared with the signature verifier. We enhance the Barrett reduction algorithm by adapting it to convert modular multiplications into divisions, utilizing the Barrett parameter $R = 2^k$, where k significantly surpasses the bit length of S_1 and S_2 . This adaptation addresses the potential issue wherein the floor function of the Barrett reduction algorithm may occasionally yield a result $z - 1$ instead of z due to the limitations of the floor function's cutoff. Then the conditional identity equation with $\pmod p$ could be partially applied to the moduli S_1 and S_2 to produce public key elements $s_1 = \beta * S_1 \pmod p$ and $s_2 = \beta * S_2 \pmod p$ with a random chosen $\beta \in \mathbb{F}_p$. The signature elements from the private polynomials $f(x)$ and $h(x)$ are calculated with $x = \text{Hash}(M)$ and then decrypted into F and H as signature elements with R_2 and R_1 respectively.

Related work is reviewed in Section 3. The HPPK-THR KEM cryptography system is briefly described in Section 4. Section 5 introduces HPPK DS. Section 6 discusses the security analysis. We conclude with Section 7.

3 Related Work

In the realm of Post-Quantum Cryptography (PQC), the field of digital signatures is currently undergoing extensive exploration and development. This is in response to the potential threats posed by quantum computers to traditional cryptographic algorithms. The National Institute of Standards and Technology (NIST) initiated the standardization

process for Post-Quantum Cryptography (PQC) in 2017 [11], initially considering 69 candidates. The first round concluded in 2019, narrowing down the field to 26 candidates entering the second round [12].

Among these candidates, only four Key Encapsulation Mechanism (KEM) options progressed to the third round: code-based Classic McEliece [13], lattice-based CRYSTALS-KYBER [14], NTRU [15], and SABER [16]. Additionally, three digital signature candidates advanced to the third round: lattice-based CRYSTALS-DILITHIUM [17] and FALCON [18], along with multivariate Rainbow [19], as shown in NIST status report in 2021 [20]. In a subsequent update, NIST declared Kyber as the sole KEM candidate for standardization [21]. For digital signatures, NIST identified CRYSTALS-Dilithium, FALCON, and SPHINCS+ [22] as the three standardized algorithms.

In early 2022, vulnerabilities in the NIST round 3 finalists came to light. Damien Robert initiated this revelation by reporting an attack on Supersingular Isogeny Diffie–Hellman (SIDH) in polynomial time [23, 24]. Subsequently, Castryck and Decru presented a more efficient key recovery attack on SIDH[25], achieving key recovery for NIST security level V in under 2 hours using a laptop.

In a separate development, a novel cryptoanalysis was introduced by Emily Wenger et al. in 2022 [26]. This approach utilizes Machine Learning (ML) for the secret recovery of lattice-based schemes. Wenger’s team demonstrated that their attack could fully recover secrets for small-to-midsize Learning With Errors (LWE) instances with sparse binary secrets, up to lattice dimensions of $n = 128$ with SALSA [26], $n = 350$ with PICANTE [27], and $n = 512$ with VERDE [28]. The potential scalability of this attack to real-world LWE-based cryptosystems is being explored. The team is actively working on enhancing the attack’s capability to target larger parameter sets, although the timeframe and resources required for this advancement remain uncertain. Their latest publishing not only improves the secret recovery methods of SALSA but also introduce a novel cross-attention recovery mechanism [28]. Their innovative attack has ushered in a new era of cryptoanalysis, particularly when integrating ML with quantum computing.

Sharp et al. recently achieved significant breakthroughs in integer factorization, as detailed in their report [29]. Their research focuses on breaking a 300-bit RSA public key using the MemComputing technique in simulation mode. MemComputing, a novel computing paradigm characterized by time non-locality, represents a noteworthy departure from traditional computing methods. In contrast to quantum computing, which relies on physical quantum processing units and exponential quantum resources for substantial computational acceleration, MemComputing demonstrates the ability to achieve large prime factorizations with polynomial computing resources. This holds true even when using classical computing systems, although further acceleration can be attained through ASIC implementation. Notably, Zhang and Ventra have recently proposed a digital implementation of MemComputing [30]. The implications of this technological advancement suggest that classical cryptography may become vulnerable sooner than anticipated, potentially outpacing the threat posed by quantum computers.

4 Brief of Homomorphic Polynomial Public Key Cryptography

In this section, we are going to briefly describe the motivation behind and development of the HPPK scheme: the path from the original deterministic polynomial public key or DPPK, proposed by Kuang in 2021 [1], then multivariate polynomial public key or MPPK, proposed by Kuang, Perepechaenko, and Barbeau in 2022 [5], to the homomorphic polynomial public key over a single hidden ring in 2023 [6], and over a dual hidden ring scheme [7]. MPPK schemes for digital signature or MPPK/DS have been proposed by Kuang, Perepechaenko, and Barbeau in 2022 [8], then an optimized version was proposed by Kuang and Perepechaenko in 2023 [9]. A forged signature attack was reported by Guo in 2023 [10].

4.1 DPPK

The DPPK cryptography was first proposed by Kuang in 2021 to leverage the elementary identity equation [1]

$$\frac{f(x)}{h(x)} = \frac{B(x)f(x)}{B(x)h(x)} = \frac{P(x)}{Q(x)} \pmod{p} \quad (2)$$

with p being a prime, $f(x)$ and $h(x) \in \mathbb{F}_p$ as private solvable polynomials, and random base polynomial $B(x) \in \mathbb{F}_p$ used to encrypt $f(x)$ and $h(x)$. They are defined as follows over the field \mathbb{F}_p ,

$$\begin{aligned} f(x) &= f_0 + f_1x + \cdots + f_\lambda x^\lambda, \\ h(x) &= h_0 + h_1x + \cdots + h_\lambda x^\lambda, \\ B(x) &= B_0 + B_1x + \cdots + B_n x^n. \end{aligned} \quad (3)$$

Product polynomials $P(x) = f(x)B(x)$ and $Q(x) = h(x)B(x)$ are used to create public key polynomials $P'(x) = P(x) - P_0$ and $Q'(x) = Q(x) - Q_0$ by excluding their constant terms P_0 and Q_0 respectively. The randomly chosen

secret $x \in \mathbb{F}_p$ is encrypted into ciphertext $\bar{P} = P'(x)$ and $\bar{Q} = Q'(x)$ and is extracted through a radical expression through following equation

$$\frac{f(x)}{h(x)} = \frac{\bar{P} + f_0 B_0}{\bar{Q} + h_0 B_0} = k \pmod{p} \longrightarrow f(x) - kh(x) = 0 \pmod{p}. \quad (4)$$

Indeed, given that the decrypting party has knowledge of values $f_0 B_0, h_0 B_0, \bar{P}, \bar{Q}$, they can calculate value k , which can then be used in the equation

$$\frac{f(x)}{h(x)} = k \pmod{p} \quad (5)$$

to find x . Although the key recovery attack on the DPPK scheme is non-deterministic, and can be carried out by solving public key equation system using Gaussian elimination technique, DPPK is vulnerable to the secret recovery attack from the ciphertext under Generalized Riemann Hypothesis or GRH by Evdokimov [2].

4.2 MPPK Key Encapsulation Mechanism

In order to overcome the DPPK vulnerability with respect to the secret recovery attack, Kuang and Barbeau in 2021 [3, 4] proposed to replace the randomly chosen base polynomial $B(x)$ with a randomly chosen multivariate polynomial $B(x, u_1, \dots, u_m)$ over the prime field \mathbb{F}_p , called MPPK KEM. The variable x denotes the secret and u_1, \dots, u_m refer to the randomly chosen noise variables from \mathbb{F}_p . The public key polynomials $P'(x, u_1, \dots, u_m)$ and $Q'(x, u_1, \dots, u_m)$ are created from the product polynomials $P(x, u_1, \dots, u_m), Q(x, u_1, \dots, u_m)$ excluding their constant terms and highest order terms. The constant terms and the highest order terms are used to create two noise functions $N_0(u_1, \dots, u_m) = R_0 B_0(u_1, \dots, u_m)$ and $N_n(x, u_1, \dots, u_m) = R_n B_n(u_1, \dots, u_m) x^{n+\lambda} \in \mathbb{F}_p$, where R_0 and R_n are the private key values, randomly selected from \mathbb{F}_p .

Key encapsulation mechanism in MPPK is achieved through the randomly chosen noise variables $u_1, \dots, u_m \in \mathbb{F}_p$. The ciphertext consists of four elements $\bar{P} = P'(x, u_1, \dots, u_m), \bar{Q} = Q'(x, u_1, \dots, u_m), \bar{N}_0 = N_0(u_1, \dots, u_m)$, and $\bar{N}_n = N_n(x, u_1, \dots, u_m)$. And the decryption is the same as in DPPK

$$k = \frac{f(x)}{h(x)} = \frac{B(x, x_1, \dots, x_m) f(x)}{B(x, x_1, \dots, x_m) h(x)} = \frac{P(x, x_1, \dots, x_m)}{Q(x, x_1, \dots, x_m)} \pmod{p}, \quad (6)$$

where polynomial values of $P(x, u_1, \dots, u_m)$ and $Q(x, u_1, \dots, u_m)$ are calculated with four elements of ciphertext and private key: f_0, h_0, R_0 , and R_n .

The two noise functions $N_0(u_1, \dots, u_m)$ and $N_n(x, u_1, \dots, u_m) \in \mathbb{F}_p$ partially leak B_{0j} and B_{nj} because of $\frac{N_{0j}}{N_{0\ell}} = \frac{R_0 B_{0j}}{R_0 B_{0\ell}} = \frac{B_{0j}}{B_{0\ell}} \pmod{p}$ and $\frac{N_{nj}}{N_{n\ell}} = \frac{R_n B_{nj}}{R_n B_{n\ell}} = \frac{B_{nj}}{B_{n\ell}}$. Kuang, Perepechaenko, and Barbeau in 2022 [5] proposed a mechanism by choosing R_0 and R_n to encrypt the coefficients of two noise functions over a hidden ring $\mathbb{Z}/S\mathbb{Z}$ with the bit length of the ring two times bigger than that of the prime field \mathbb{F}_p . Having the value S hidden, implies that S is a part of the private key. Without knowing S , the divisions $\frac{N_{0j}}{N_{0\ell}}$ and $\frac{N_{nj}}{N_{n\ell}}$ can not be carried out successfully. This eliminates the vulnerability originally present in the two noise functions.

The ciphertext of MPPK KEM with a hidden ring consist of elements but the elements \bar{N}_0 and \bar{N}_n are three times bigger than the size of the prime field.

4.3 HPPK KEM

In the framework of the MPPK KEM scheme, the security of all coefficients within two noise functions was safeguarded through encryption over a hidden ring with a bit length $2 \times$ larger than the prime field bit length. This concept was subsequently expanded to encompass all coefficients of product polynomials $P(x, u_1, \dots, u_m)$ and $Q(x, u_1, \dots, u_m)$ by Kuang and Perepechaenko in 2023 [6]. The encryption method has been demonstrated to be partially homomorphic, under addition and scalar multiplication operations. Consequently, the scheme was renamed as Homomorphic Polynomial Public Key, or HPPK.

In HPPK KEM, the entire product polynomials $P(x, u_1, \dots, u_m)$ and $Q(x, u_1, \dots, u_m)$ are considered private

$$\begin{aligned} P(x, u_1, \dots, u_m) &= f(x)B(x, u_1, \dots, u_m) = \sum_{j=1}^m \sum_{i=0}^{n+\lambda} p_{ij} x^i u_j \pmod{p} \\ Q(x, u_1, \dots, u_m) &= h(x)B(x, u_1, \dots, u_m) = \sum_{j=1}^m \sum_{i=0}^{n+\lambda} q_{ij} x^i u_j \pmod{p} \end{aligned} \quad (7)$$

with $p_{ij} = \sum_{s+t=i} f_s B_{tj} \bmod p$ and $q_{ij} = \sum_{s+t=i} h_s B_{tj} \bmod p$. Choosing a secret S with its bit length to be more than double of the prime field \mathbb{F}_p and secret encryption key R_1 and R_2 over the ring with condition $\gcd(R_1, S) = 1$ and $\gcd(R_2, S) = 1$, we then encrypt entire coefficients of $P(\cdot)$ and $Q(\cdot)$ as follows

$$p'_{ij} = R_1 \times p_{ij} \bmod S, q'_{ij} = R_2 \times q_{ij} \bmod S \quad (8)$$

then the cipher polynomials can be written as

$$\begin{aligned} \mathcal{P}(x, u_1, \dots, u_m) &= \sum_{j=1}^m \sum_{i=0}^{n+\lambda} p'_{ij} (x^i u_j \bmod p) \\ \mathcal{Q}(x, u_1, \dots, u_m) &= \sum_{j=1}^m \sum_{i=0}^{n+\lambda} q'_{ij} (x^i u_j \bmod p) \end{aligned} \quad (9)$$

The secret encryption is the same as in MPPK KEM, but now we only have two elements for the ciphertext: \bar{P}, \bar{Q} . The decryption is split into two phases: symmetric homomorphic decryption with R_1 and R_2

$$\bar{P} = (R_1^{-1} \times \bar{P} \bmod S) \bmod p, \bar{Q} = (R_2^{-1} \times \bar{Q} \bmod S) \bmod p \quad (10)$$

and then the secret extraction which is the same as in DPPK. In the most recent iteration of HPPK KEM, Kuang and Perepechaenko (2023) propose employing two distinct rings, $\mathbb{Z}/S_1\mathbb{Z}$ for the encryption of $P(\cdot)$ and $\mathbb{Z}/S_2\mathbb{Z}$ for the encryption of $Q(\cdot)$, to further enhance the security [7].

4.4 MPPK DS

A digital signature scheme based on MPPK has been proposed by Kuang, Perepechaenko, and Barbeau in 2022 [8] and later optimized variant was proposed by Kuang and Perepechaenko in 2023 [9]. MPPK digital signature scheme originated from MPPK over a single prime field [3, 4]. The signature verification equation stems from the identity equation

$$f(x)Q(x, u_1, \dots, u_m) = h(x)P(x, u_1, \dots, u_m) \bmod \varphi(p), \quad (11)$$

where $\varphi(p)$ is an Euler's totient function. For a prime p , $\varphi(p) = p - 1$. With the inclusion of two noise functions $N_0(u_1, \dots, u_m) = R_0 B_0(u_1, \dots, u_m)$ and $N_n(x, u_1, \dots, u_m) = R_n B_n(u_1, \dots, u_m) x^{n+\lambda} \in \mathbb{F}_p$, Eq. (11) becomes

$$\begin{aligned} f(x)Q'(x, u_1, \dots, u_m) &= h(x)P'(x, u_1, \dots, u_m) + s_0(x)N_0(u_1, \dots, u_m) \\ &\quad + s_n(x)N_n(x, u_1, \dots, u_m) \bmod \varphi(p). \end{aligned} \quad (12)$$

with $s_0(x) = f_0 h(x) - h_0 f(x)$, $s_n(x) = f_\lambda h(x) - h_\lambda f(x)$. Choosing a secret base $g \in \mathbb{F}_p$ to avoid the discrete logarithm and performing modular exponentiation to the above Eq. (12), we obtain

$$A^{Q'(x, u_1, \dots, u_m)} = B^{P'(x, u_1, \dots, u_m)} C^{N_0(u_1, \dots, u_m)} D^{N_n(x, u_1, \dots, u_m)} \bmod p \quad (13)$$

with $A = g^{f(x)}$, $B = g^{h(x)}$, $C = g^{s_0(x)}$, $D = g^{s_n(x)}$ as signature elements. Recently, Guo uncovered a forged signature attack on MPPK DS [10], attributed to the linearity between signature elements and public polynomials of Eq. (12). Attempts were made to address this vulnerability in MPPK DS, but it proved exceedingly challenging to rectify within a single modulo domain due to the linearity.

5 HPPK Digital Signature Scheme

The homomorphic encryption operator $\hat{\mathcal{E}}_{(R,S)}$ was defined in [6, 7] as

$$\hat{\mathcal{E}}_{(R,S)}(f) = (R \circ f) \bmod S = f', \quad (14)$$

with f being a polynomial and decryption operator $\hat{\mathcal{E}}_{(R,S)}^{-1}$

$$\hat{\mathcal{E}}_{(R,S)}^{-1}(f') = (R^{-1} \circ f') \bmod S. \quad (15)$$

So it is very easy to verify that

$$\hat{\mathcal{E}}_{(R,S)}^{-1}(f') = \hat{\mathcal{E}}_{(R,S)}^{-1} \hat{\mathcal{E}}_{(R,S)}(f) = (R^{-1} \times R \bmod S) \circ f = f. \quad (16)$$

which reveals the unitary and reversible relation

$$\hat{\mathcal{E}}_{(R,S)}^{-1} \hat{\mathcal{E}}_{(R,S)} = 1. \quad (17)$$

Indeed, this encryption operator serves as a distinctive permutation operator or a quantum permutation gate that can be implemented in a native quantum computing system [31, 32]. Furthermore, it adheres to the principles of non-commutativity, i.e.

$$\hat{\mathcal{E}}_{(R,S)} \hat{\mathcal{E}}_{(R',S')} \neq \hat{\mathcal{E}}_{(R',S')} \hat{\mathcal{E}}_{(R,S)}. \quad (18)$$

The non-commutativity is very important for us to reuse it multiple times to encrypt the entire coefficients of a product polynomial without reducing its security [33], unlike the case of One-Time-Pad encryption with XOR because XOR operator is commutable.

5.1 HPPK DS Algorithm

Let's insert Eq. (17) into Eq. (11) with a randomly chosen $\alpha \in \mathbb{F}_p$

$$\{[\alpha f(x) \bmod p][\hat{\mathcal{E}}_{(R_2,S_2)}^{-1} \hat{\mathcal{E}}_{(R_2,S_2)}]Q(\dots)\} \bmod p = \{[\alpha h(x) \bmod p][\hat{\mathcal{E}}_{(R_1,S_1)}^{-1} \hat{\mathcal{E}}_{(R_1,S_1)}]P(\dots)\} \bmod p \quad (19)$$

then we reorganize Eq. (19) into a verification equation

$$F(x)Q(x, u_1, \dots, u_m) \bmod p = H(x)P(x, u_1, \dots, u_m) \bmod p \quad (20)$$

with

$$F(x) = \hat{\mathcal{E}}_{(R_2,S_2)}^{-1}[\alpha f(x) \bmod p] = R_2^{-1} \times [\alpha f(x) \bmod p] \bmod S_2 \quad (21)$$

$$H(x) = \hat{\mathcal{E}}_{(R_1,S_1)}^{-1}[\alpha h(x) \bmod p] = R_1^{-1} \times [\alpha h(x) \bmod p] \bmod S_1 \quad (22)$$

as signature polynomials with $\alpha \in \mathbb{F}_p$ randomly chosen per signing message and

$$\begin{aligned} \mathcal{P}(x, u_1, \dots, u_m) &= \hat{\mathcal{E}}_{(R_1,S_1)} P(x, u_1, \dots, u_m) \\ &= \left\{ \sum_{j=1}^m \sum_{i=0}^{n+\lambda} [R_1 \times p_{ij} \bmod S_1] (x^i u_j \bmod p) \right\} \bmod S_1 \\ &= \left\{ \sum_{j=1}^m \sum_{i=0}^{n+\lambda} P_{ij} (x^i u_j \bmod p) \right\} \bmod S_1 \end{aligned} \quad (23)$$

$$\begin{aligned} \mathcal{Q}(x, u_1, \dots, u_m) &= \hat{\mathcal{E}}_{(R_2,S_2)} Q(x, u_1, \dots, u_m) \\ &= \left\{ \sum_{j=1}^m \sum_{i=0}^{n+\lambda} [R_2 \times q_{ij} \bmod S_2] (x^i u_j \bmod p) \right\} \bmod S_2 \\ &= \left\{ \sum_{j=1}^m \sum_{i=0}^{n+\lambda} Q_{ij} (x^i u_j \bmod p) \right\} \bmod S_2 \end{aligned} \quad (24)$$

as homomorphically encrypted polynomials with coefficients $P_{ij} = R_1 * p_{ij} \bmod S_1$ and $Q_{ij} = R_2 * q_{ij} \bmod S_2$. We can then rewrite the verification equation Eq. (20) as

$$\begin{aligned} \sum_{j=1}^m \sum_{i=0}^{n+\lambda} [F(x)Q_{ij} \bmod S_2] (x^i u_j \bmod p) \bmod p &= \sum_{j=1}^m \sum_{i=0}^{n+\lambda} [H(x)P_{ij} \bmod S_1] (x^i u_j \bmod p) \bmod p \\ \longrightarrow \sum_{j=1}^m \sum_{i=0}^{n+\lambda} V_{ij}(F) x^i u_j \bmod p &= \sum_{j=1}^m \sum_{i=0}^{n+\lambda} U_{ij}(H) x^i u_j \bmod p \\ \longrightarrow V(F, x, u_1, \dots, u_m) &= U(H, x, u_1, \dots, u_m) \bmod p \end{aligned} \quad (25)$$

with polynomial coefficients $V_{ij}(F)$ and $U_{ij}(H)$ to be defined as

$$\begin{aligned} U_{ij}(H) &= [H(x)P_{ij} \bmod S_1] \bmod p \\ V_{ij}(F) &= [F(x)Q_{ij} \bmod S_2] \bmod p. \end{aligned} \quad (26)$$

The coefficients newly defined as $V_{ij}(F)$ and $U_{ij}(H)$ in Eq.(26) cannot be directly computed due to the hidden nature of moduli S_1 and S_2 , as they are not accessible to the verifier. However, the Barrett reduction algorithm offers a solution, enabling the transformation of modular multiplications over both hidden rings into divisions[34]:

$$ab \bmod n = ab - n \lfloor \frac{ab}{n} \rfloor = ab - n \lfloor \frac{a \lfloor \frac{Rb}{n} \rfloor}{R} \rfloor = ab - n \lfloor \frac{a\mu}{R} \rfloor \quad (27)$$

with $\mu = \lfloor \frac{Rb}{n} \rfloor$ as the Barrett parameter and $R = 2^k$, $k \geq \lceil \log_2 n \rceil$, the Barrett reduction algorithm proves instrumental in enhancing the efficiency of modular multiplications with the precomputed μ . The outcome z generated by the Barrett algorithm typically falls within the range of $[0, 2n)$ but not consistently within $[0, n)$. Therefore, if the result exceeds n , it necessitates returning $z - n$. Notably, it has been observed that the occurrence of $z > n$ cases can be mitigated by increasing the bit length of R . Our testing indicates that when $k - \lceil \log_2 n \rceil > 30$, the instances of $z > n$ tend to approach zero after 100,000,000 trials.

With Eq. (27), we can transform Eq. (26) into the following equations by multiplying a randomly chosen $\beta \in \mathbb{F}_p$ and then taking mod p

$$\begin{aligned} U_{ij}(H) &= H(x)p'_{ij} - s_1 \lfloor \frac{H(x)\mu_{ij}}{R} \rfloor \bmod p \\ V_{ij}(F) &= F(x)q'_{ij} - s_2 \lfloor \frac{F(x)\nu_{ij}}{R} \rfloor \bmod p. \end{aligned} \quad (28)$$

with

$$\begin{aligned} s_1 &= \beta S_1 \bmod p \\ s_2 &= \beta S_2 \bmod p \\ p'_{ij} &= \beta P_{ij} \bmod p \\ q'_{ij} &= \beta Q_{ij} \bmod p \\ \mu_{ij} &= \lfloor \frac{RP_{ij}}{S_1} \rfloor \\ \nu_{ij} &= \lfloor \frac{RQ_{ij}}{S_2} \rfloor \end{aligned} \quad (29)$$

to be elements of the **public key** for HPPK DS. The hidden S_1, S_2 are no longer required for the signature verification. The **private key** consists of:

$$\begin{aligned} f(x), h(x) &: h_i, i = 0, 1, \dots, \lambda \\ R_1 &\in \mathbb{Z}/S_1\mathbb{Z} \\ R_2 &\in \mathbb{Z}/S_2\mathbb{Z} \end{aligned} \quad (30)$$

It should be pointed out that coefficients in Eq. (28) of public polynomials in Eq. (25) are non-linearly associated with signature elements $F(x)$ and $H(x)$ at variable value x (note, please treat $F(x)$ and $H(x)$ as numbers of polynomial values at message x), or called signature embedded coefficients. This non-linearity prevents the forged signature found in our early MPPK DS scheme [10]. We will delve into this aspect in the subsequent security analysis section, where we set $\beta = 1$ for the independence of s_1 and s_2 , alongside a randomly selected value governing the association between s_1 and s_2 .

5.2 Signing

The signing algorithm is very straightforward in HPPK DS:

- Assign the hash code of a message to variable $x = Hash(M)$;
- Randomly choose $\alpha \in \mathbb{F}_p$;
- Evaluate $F = R_2^{-1} \times [\alpha f(x) \bmod p] \bmod S_2$;
- Evaluate $H = R_1^{-1} \times [\alpha h(x) \bmod p] \bmod S_1$;
- HPPK DS $sig = \{F, H\}$.

Subsequently, the signer appends the signature $sig = \{F, H\}$ to the message, possibly alongside the public key if the recipient lacks it. The recipient of the message can then verify the signature using the public key of the HPPK DS. In

cases where the hash code of the message exceeds the bit length of the finite field \mathbb{F}_p , segmentation becomes necessary, aligning with \mathbb{F}_p , and the segmented parts are concatenated to form the message's signature.

To prevent signature verification failure, it is advisable to conduct verification, as outlined in subsection 5.3. In the event of verification failure, it is recommended to randomly select a new $\alpha \in \mathbb{F}_p$ and reevaluate F and H .

5.3 Verify

The signature verification consists of two stages: evaluating signature embedded coefficients and comparing polynomial values at the variable value x . The procedure is as follows

- $x = Hash(M)$, and randomly choose variables $u_1, \dots, u_m \in \mathbb{F}_p$;
- Evaluate $U_{ij}(H), V_{ij}(F)$ based on Eq. (28) for $i = 0, \dots, n + \lambda, j = 1, \dots, m$;
- Evaluate $V(F, x, u_1, \dots, u_m)$ and $U(H, x, u_1, \dots, u_m)$ and check if they are equal. If they are equal, the verification is passed. Otherwise, the verification is failed.

5.4 A Variant of the Barrett Reduction Algorithm

The Barrett reduction algorithm [34] is employed to enhance the efficiency of modular operations, particularly modular multiplications as expressed in Eq. (27). The outcome z typically lies within the range of $[0, 2n)$ due to the limitations imposed by the pre-computed floor function $\lfloor \frac{Rb}{n} \rfloor$ with $R = 2^k$. In the context of the proposed HPPK DS scheme, however, it is essential for the result to fall within $[0, n)$. Notably, by increasing the value of k , the results obtained from the Barrett reduction algorithm can be effectively constrained to the desired range of $[0, n)$.

All three lines exhibit a roughly linear relationship in the semi-log graph, converging to zero as δ surpasses 24 bits. However, as k decreases to $\lceil \log_2 n \rceil$, the instances of falling beyond $[0, n)$ rise to 6% for $\lceil \log_2 n \rceil = 208$ bits, 2% for $\lceil \log_2 n \rceil = 292$ bits, and 0.3% for $\lceil \log_2 n \rceil = 400$ bits, respectively. Consequently, achieving a result within $[0, n)$ with high probability is feasible when $\delta = k - \lceil \log_2 n \rceil > 32$ bits. In Fig. 1, a semi-logarithmic graph depicts the count of Barrett reduction results ($z = a * b \bmod n$) falling within the range $[n, 2n)$ per 10^8 computations, where $a < n$ and $b < n$ are randomly chosen. The graph is plotted against the parameter $\delta = k - \lceil \log_2 n \rceil$. Three scenarios are represented by the blue line for $\lceil \log_2 n \rceil = 208$ bits, the yellow line for $\lceil \log_2 n \rceil = 304$ bits, and the grey line for $\lceil \log_2 n \rceil = 400$ bits, respectively.

5.5 A Toy Example

5.5.1 Key Pair Generation

For purposes of simplicity, consider HPPK DS defined over a prime field \mathbb{F}_{13} and choose $S_1 = 6797, S_2 = 7123$. we select $R_1 = 4267, R_2 = 6475$. Let the randomly chosen private key consist of the following values:

$$\begin{cases} S_1 = 6797, R_1 = 4267 \\ S_2 = 7123, R_2 = 6475 \\ f(x) = 4 + 9x \\ h(x) = 10 + 7x \end{cases} \quad (31)$$

Let the base polynomial $B(\cdot)$ randomly generated for this example be

$$B(x, u_1, u_2) = (8 + 7x)u_1 + (5 + 11x)u_2$$

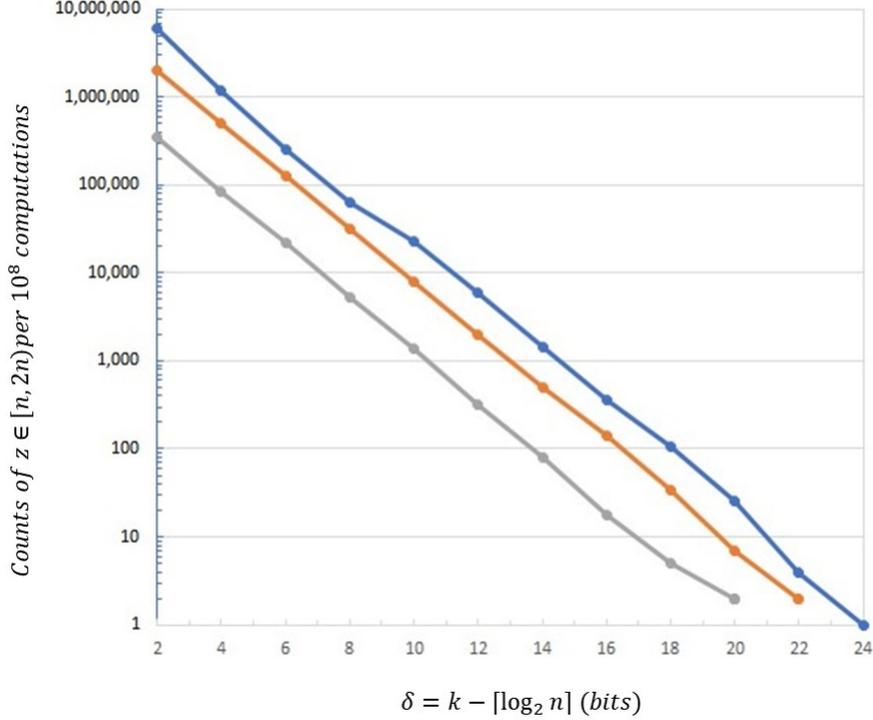
and then product polynomials

$$\begin{aligned} P(x, u_1, u_2) &= f(x)B(x, u_1, u_2) = (6 + 9x + 11x^2)u_1 + (7 + 11x + 8x^2)u_2 \bmod 13 \\ Q(x, u_1, u_2) &= h(x)B(x, u_1, u_2) = (2 + 9x + 10x^2)u_1 + (11 + 2x + 12x^2)u_2 \bmod 13 \end{aligned}$$

Now let's encryption $P(x)$ and $Q(x)$ with R_1 over the rings

$$\begin{aligned} \mathcal{P}(x, u_1, u_2) &= 4267 * P(x, u_1, u_2) \bmod 6797 \\ &= (5211 + 4418x + 6155x^2)u_1 + (2681 + 6155x + 151x^2)u_2 \\ \mathcal{Q}(x, u_1, u_2) &= 6475 * Q(x, u_1, u_2) \bmod 7123 \\ &= (5827 + 1291x + 643x^2)u_1 + (7118 + 5827x + 6470x^2)u_2 \end{aligned}$$

Figure 1: A semi-log graph illustration of the Barrett reduction results falling in $[n, 2n)$ per 10^8 computations of $z = a * b \bmod n$, with randomly chosen $a < n$ and $b < n$, is plotted as a function of $\delta = k - \lceil \log_2 n \rceil$ in bits. The blue line corresponds to $\lceil \log_2 n \rceil = 208$ bits, the yellow line to $\lceil \log_2 n \rceil = 292$ bits, and the grey line to $\lceil \log_2 n \rceil = 400$ bits.



Then we obtain the key pair by choosing the Barrett parameter $R = 2^{24}$.

Private Key:

$$f_0 = 4, f_1 = 9; h_0 = 10, h_1 = 9$$

$$R_1 = 4267, S_1 = 6797; R_2 = 6475, S_2 = 7123$$

Public Key: For simplification, we choose $\beta = 1$ to evaluate all public key elements.

$$s_1 = S_1 \bmod 13 = 11, s_2 = S_2 \bmod 13 = 12$$

$$p'_{01} = P_{01} \bmod 13 = 5211 \bmod 13 = 11$$

$$p'_{11} = P_{11} \bmod 13 = 4418 \bmod 13 = 11$$

$$p'_{21} = P_{21} \bmod 13 = 6155 \bmod 13 = 6$$

$$p'_{02} = P_{02} \bmod 13 = 2681 \bmod 13 = 3$$

$$p'_{12} = P_{12} \bmod 13 = 6155 \bmod 13 = 6$$

$$p'_{22} = P_{22} \bmod 13 = 151 \bmod 13 = 8$$

$$q'_{01} = Q_{01} \bmod 13 = 5827 \bmod 13 = 3$$

$$\begin{aligned}
q'_{11} &= Q_{11} \bmod 13 = 1291 \bmod 13 = 4 \\
q'_{21} &= Q_{21} \bmod 13 = 643 \bmod 13 = 6 \\
q'_{02} &= Q_{02} \bmod 13 = 7118 \bmod 13 = 7 \\
q'_{12} &= Q_{12} \bmod 13 = 5827 \bmod 13 = 3 \\
q'_{22} &= Q_{22} \bmod 13 = 6470 \bmod 13 = 9 \\
\mu'_{01} &= \lfloor \frac{RP_{01}}{S_1} \rfloor = \lfloor \frac{2^{24} * 5211}{6797} \rfloor = 12862449 \\
\mu'_{11} &= \lfloor \frac{RP_{11}}{S_1} \rfloor = \lfloor \frac{2^{24} * 4418}{6797} \rfloor = 10905066 \\
\mu'_{21} &= \lfloor \frac{RP_{21}}{S_1} \rfloor = \lfloor \frac{2^{24} * 6155}{6797} \rfloor = 15192550 \\
\mu'_{02} &= \lfloor \frac{RP_{02}}{S_1} \rfloor = \lfloor \frac{2^{24} * 2681}{6797} \rfloor = 6617583 \\
\mu'_{12} &= \lfloor \frac{RP_{12}}{S_1} \rfloor = \lfloor \frac{2^{24} * 6155}{6797} \rfloor = 15192550 \\
\mu'_{22} &= \lfloor \frac{RP_{22}}{S_1} \rfloor = \lfloor \frac{2^{24} * 151}{6797} \rfloor = 372717 \\
\nu'_{01} &= \lfloor \frac{RQ_{01}}{S_2} \rfloor = \lfloor \frac{2^{24} * 5827}{7123} \rfloor = 13724671 \\
\nu'_{11} &= \lfloor \frac{RQ_{11}}{S_2} \rfloor = \lfloor \frac{2^{24} * 1291}{7123} \rfloor = 3040767 \\
\nu'_{21} &= \lfloor \frac{RQ_{21}}{S_2} \rfloor = \lfloor \frac{2^{24} * 643}{7123} \rfloor = 1514495 \\
\nu'_{02} &= \lfloor \frac{RQ_{02}}{S_2} \rfloor = \lfloor \frac{2^{24} * 7118}{7123} \rfloor = 16765439 \\
\nu'_{12} &= \lfloor \frac{RQ_{12}}{S_2} \rfloor = \lfloor \frac{2^{24} * 5827}{7123} \rfloor = 13724671 \\
\nu'_{22} &= \lfloor \frac{RQ_{22}}{S_2} \rfloor = \lfloor \frac{2^{24} * 6470}{7123} \rfloor = 15239167
\end{aligned}$$

The above public key together with security parameters $p = 13, R = 2^{24}$ would be available for any verifier to process any signature generated by the true signer holding the private key.

5.5.2 Signing

Assume that the hashed value of a message M is $x = 9$. Here are steps to generate the signature:

- Choose a random $\alpha = 1 \in \mathbb{F}_{13}$,
- Evaluate $F = R_2^{-1} \times [\alpha f(x) \bmod p] \bmod S_2 = 6475^{-1} \times [4 + 9 * 9 \bmod 13] \bmod 7123 = 5683$
- Evaluate $H = R_1^{-1} \times [\alpha h(x) \bmod p] \bmod S_1 = 4267^{-1} \times [10 + 7 * 9 \bmod 13] \bmod 6797 = 5357$

then the signature is $sig = \{5683, 5357\}$.

5.5.3 Verify

After receiving the signature from the sender, the signature verification consists of two steps: evaluating the coefficients of verification polynomials $U(H, x, u_1, u_2)$ and $V(F, x, u_1, u_2)$ with the signature elements and then polynomial values

at the variable $x = Hash(M)$. Let's first prepare all coefficients:

$$\begin{aligned}
U_{01} &= Hp'_{01} - s_1 \lfloor \frac{H\mu_{01}}{R} \rfloor = 5357 * 11 - 11 * \lfloor \frac{5357 * 12862449}{2^{24}} \rfloor \bmod 13 = 9 \\
U_{11} &= Hp'_{11} - s_1 \lfloor \frac{H\mu_{11}}{R} \rfloor = 5357 * 11 - 11 * \lfloor \frac{5357 * 10905066}{2^{24}} \rfloor \bmod 13 = 7 \\
U_{21} &= Hp'_{21} - s_1 \lfloor \frac{H\mu_{21}}{R} \rfloor = 5357 * 6 - 11 * \lfloor \frac{5357 * 15192550}{2^{24}} \rfloor \bmod 13 = 10 \\
U_{02} &= Hp'_{02} - s_1 \lfloor \frac{H\mu_{02}}{R} \rfloor = 5357 * 3 - 11 * \lfloor \frac{5357 * 6617583}{2^{24}} \rfloor \bmod 13 = 4 \\
U_{12} &= Hp'_{12} - s_1 \lfloor \frac{H\mu_{12}}{R} \rfloor = 5357 * 6 - 11 * \lfloor \frac{5357 * 1519255}{2^{24}} \rfloor \bmod 13 = 10 \\
U_{22} &= Hp'_{22} - s_1 \lfloor \frac{H\mu_{22}}{R} \rfloor = 5357 * 8 - 11 * \lfloor \frac{5357 * 372717}{2^{24}} \rfloor \bmod 13 = 12 \\
V_{01} &= Fq'_{01} - s_2 \lfloor \frac{F\nu_{01}}{R} \rfloor = 5683 * 3 - 12 * \lfloor \frac{5683 * 13724671}{2^{24}} \rfloor \bmod 13 = 1 \\
V_{11} &= Fq'_{11} - s_2 \lfloor \frac{F\nu_{11}}{R} \rfloor = 5683 * 4 - 12 * \lfloor \frac{5683 * 3040767}{2^{24}} \rfloor \bmod 13 = 11 \\
V_{21} &= Fq'_{21} - s_2 \lfloor \frac{F\nu_{21}}{R} \rfloor = 5683 * 6 - 12 * \lfloor \frac{5683 * 3040767}{2^{24}} \rfloor \bmod 13 = 5 \\
V_{02} &= Fq'_{02} - s_2 \lfloor \frac{F\nu_{02}}{R} \rfloor = 5683 * 7 - 12 * \lfloor \frac{5683 * 16765439}{2^{24}} \rfloor \bmod 13 = 12 \\
V_{12} &= Fq'_{12} - s_2 \lfloor \frac{F\nu_{12}}{R} \rfloor = 5683 * 3 - 12 * \lfloor \frac{5683 * 13724671}{2^{24}} \rfloor \bmod 13 = 1 \\
V_{22} &= Fq'_{22} - s_2 \lfloor \frac{F\nu_{22}}{R} \rfloor = 5683 * 9 - 12 * \lfloor \frac{5683 * 15239167}{2^{24}} \rfloor \bmod 13 = 6
\end{aligned}$$

So we obtain the verification polynomials

$$\begin{aligned}
U(H = 5357, x, u_1, u_2) &= (9 + 7x + 10x^2)u_1 + (4 + 10x + 12x^2)u_2 \bmod 13 \\
V(F = 5683, x, u_1, u_2) &= (1 + 11x + 5x^2)u_1 + (12 + x + 6x^2)u_2 \bmod 13
\end{aligned} \tag{32}$$

It is easy to verify that $U_1(x = 9) = V_1(x = 9) = 11 \bmod 13$ and $U_2(x = 9) = V_2(x = 9) = 0 \bmod 13$. For any randomly chosen $u_1, u_2 \in \mathbb{F}_{13}$, $U(F = 5683, x = 9, u_1, u_2) = V(H = 5357, x = 9, u_1, u_2)$. The verification is passed.

It is noticed that the signature of a given variable value x would be randomized by choosing a random factor $\alpha \in \mathbb{F}_p$, which would lead to "random" verification polynomials as shown in Eq. (32). However, a genuine signature would pass the verification.

This toy example also demonstrates that $U_1(x) = V_1(x) \bmod 13 \rightarrow 5x^2 + 9x + 8 = 5(x + 3)(x + 4) = 0 \bmod 13$. There are two roots $r_1 = 9$ and $r_2 = 10$ satisfying the equation. That means, the same signature $\{F = 5683, H = 5357\}$ would pass the verification for $x = 9$ and $x = 10$, which leads a potential forged signature with $x = 10$. Considering $U_2(x) = V_2(x) \bmod 13 \rightarrow 6x^2 + 9x + 5 = 6(x + 4)^2 = 0 \bmod 13$ which only has a single root $r_1 = r_2 = 9 \in \mathbb{F}_{13}$. That means, $x = 10$ would not satisfy the verification $U_2(x) = V_2(x) \bmod 13$. Therefore, the minimum requirement is to set $m = 2$ to avoid potential forged signature.

6 HPPK DS Security Analysis

In this section, we present the attacks we have discovered on the HPPK DS scheme to this day, and their classical computational complexities. We consider two main attack avenues, namely key recovery and signature forgery.

6.1 Private Key Attacks

Proposition 1. *There exists a private key recovery attack on the HPPK DS scheme with classical computational complexity of $\mathcal{O}(2p(S_1 + S_2))$, where p is a prime value security parameter as in Section 4.1 with s_1 independent from s_2 in the public key, and S_1, S_2 are the hidden ring values.*

Proof. From the description of the HPPK DS scheme given in Section 5, we can deduce that $Fq'_{kj} - s_2 \lfloor \frac{F\nu_{kj}}{R} \rfloor \bmod p = FQ_{kj} \bmod S_2 \bmod p$ for any fixed choice of $k \in \{0, \dots, \lambda + n\}, j \in \{1, \dots, m\}$. The adversary can obtain the numerical value of the left-hand side expression, namely $Fq'_{kj} - s_2 \lfloor \frac{F\nu_{kj}}{R} \rfloor \bmod p$, by intercepting honest signature value F and having access to the public key elements. Thus, the adversary can obtain the value $FQ_{kj} \bmod S_2 \bmod p$ for any fixed $k \in \{0, \dots, \lambda + n\}, j \in \{1, \dots, m\}$. For simplicity, let $k = 0$. The value $FQ_{0j} \bmod S_2 \bmod p$ can also be expressed as

$$FQ_{0j} \bmod S_2 \bmod p = \alpha f(x)h_0b_{0j}.$$

In a similar manner, $HP_{kj} \bmod S_1 \bmod p = Hp'_{kj} - s_1 \lfloor \frac{H\mu_{kj}}{R} \rfloor \bmod p$, whose value can be obtained by the attacker using honest signature elements and the public key values for any fixed $k \in \{0, \dots, \lambda + n\}, j \in \{1, \dots, m\}$. The value $HP_{0j} \bmod S_1 \bmod p$ can be expressed as

$$HP_{0j} \bmod S_1 \bmod p = \alpha h(x)f_0b_{0j},$$

for any $j \in \{1, \dots, m\}$.

Suppose that the adversary chose $k = 0$. We showed that the attacker has ability to create values $\alpha h(x)f_0b_{0j} \bmod p$ and $\alpha f(x)h_0b_{0j} \bmod p$ for any fixed value $j \in \{1, \dots, m\}$. They can then consider the ratios of the form

$$\frac{\alpha f(x)h_0b_{0j}}{\alpha h(x)f_0b_{0j}} \bmod p = \frac{f(x)h_0}{h(x)f_0} \bmod p$$

for any $j \in \{1, \dots, m\}$. Suppose the value of the ratio under consideration is $K \in \mathbb{Z}/p\mathbb{Z}$. Then the equation becomes

$$\frac{f_0h_0 + f_1h_0x + \dots + f_\lambda h_0x^\lambda}{h_0f_0 + h_1f_0x + \dots + h_\lambda f_0x^\lambda} = K \bmod p$$

which can then be written as

$$f_0h_0 + f_1h_0x + \dots + f_\lambda h_0x^\lambda = Kh_0f_0 + Kh_1f_0x + \dots + Kh_\lambda f_0x^\lambda \bmod p$$

or equivalently

$$1 + \frac{f_1}{f_0}x + \dots + \frac{f_\lambda}{f_0}x^\lambda = K + K\frac{h_1}{h_0}x + \dots + K\frac{h_\lambda}{h_0}x^\lambda \bmod p.$$

The adversary can create a system of such equations for various values of message variable x and accordingly various intercepted signature values F, H . This system can be solved for ratios of the form $\frac{f_1}{f_0}, \dots, \frac{f_\lambda}{f_0}, \dots, \frac{h_1}{h_0}, \dots, \frac{h_\lambda}{h_0} \bmod p$.

The adversary can then get values $f_1, \dots, f_\lambda, h_1, \dots, h_\lambda \bmod p$ if they first obtain f_0, h_0 using a brute force search. Classical computational complexity of this step of the attack is then $\mathcal{O}(2p)$.

The adversary still needs values S_1, S_2 . Classical computational complexity of the brute force search for values S_1, S_2 is $\mathcal{O}(S_1 + S_2)$. Having values S_1, S_2 the adversary can find value $\alpha R_1^{-1} = \frac{F}{f(x)}, \alpha R_2^{-1} = \frac{H}{f(x)}$ for an honest intercepted signature elements F, H . Having all of the obtained values, the adversary would be able to forge signatures for any message x . This brings the overall classical computational complexity of this attack to $\mathcal{O}(2p(S_1 + S_2)) = \mathcal{O}(2p(S_1 + S_2))$. \square

An improved version of this attack is proposed in Proposition 2 below.

Proposition 2. *There exists a private recovery attack on the HPPK DS scheme with classical computational complexity of $\mathcal{O}(p(\frac{S_1}{p} * \frac{S_2}{p}))$, where p is a prime value security parameter as in Section 4.1 with s_1 depending on s_2 in the public key 29.*

Proof. As stated in Section 5, the values s_1, s_2 are publicly shared as a part of the public key. The values s_1, s_2 are such that $s_1 = \beta S_1 \bmod p$ and $s_2 = \beta S_2 \bmod p$. Thus, they can be expressed as $S_1 = \frac{s_1}{\beta} + kp$ and $S_2 = \frac{s_2}{\beta} + tp$, for some positive integers k, t . Given bit-length of S_1, S_2 , we can infer that $k \leq \frac{S_1}{p}$ and $t \leq \frac{S_2}{p}$. Using brute force search, the adversary can find values β, k, t , and hence can find values S_1, S_2 . The classical computational complexity of this step of the attack is $\mathcal{O}(p(\frac{S_1}{p} * \frac{S_2}{p}))$.

We will now show that having found values S_1, S_2 it is possible to forge a signature in the framework of HPPK DS scheme. Indeed, consider the expression $U_{0j} = [H(x)P_{0j} \bmod S_1 \bmod p]$, for some $j \in \{1, \dots, m\}$. The adversary can recreate the value of this expression using intercepted signature values and elements of the public key.

This expression, however, is also equal to $\alpha h(x) f_0 b_{0j} \pmod p$ for the same value of $j \in \{1, \dots, m\}$. Similarly, the value V_{0j} is equal to $\alpha f(x) h_0 b_{0j} \pmod p$ for the same $j \in \{1, \dots, m\}$. The adversary can then consider ratio of the form

$$\frac{\alpha f(x) h_0 b_{0j}}{\alpha h(x) f_0 b_{0j}} \pmod p = \frac{f(x) h_0}{h(x) f_0} \pmod p = W,$$

for that same fixed j . This equation can be expressed as

$$1 + \frac{f_1}{f_0} x + \dots + \frac{f_\lambda}{f_0} x^\lambda = W + W \frac{h_1}{h_0} x + \dots + W \frac{h_\lambda}{h_0} x^\lambda \pmod p.$$

The adversary can generate more equations of this form for various intercepted values of F, H , and corresponding different values x . Having more of these equations, the adversary can solve the system of these equations for values $\frac{f_1}{f_0}, \dots, \frac{f_\lambda}{f_0}, \frac{h_1}{h_0}, \dots, \frac{h_\lambda}{h_0}$.

For an honest intercepted signature F and it's corresponding message x , the adversary can now evaluate the expression $1 + \frac{f_1}{f_0} x + \dots + \frac{f_\lambda}{f_0} x^\lambda \pmod p = E$. The adversary can also evaluate $\frac{F}{E} \pmod{S_2} = R_2^{-1} \alpha f_0 \pmod{S_2} = R_2' \pmod{S_2}$. Similarly the adversary can obtain $R_1' = \frac{H}{1 + \frac{h_1}{h_0} x + \dots + \frac{h_\lambda}{h_0} x^\lambda \pmod p} \pmod{S_1} = R_1^{-1} \alpha h_0 \pmod{S_1}$. Using these values the adversary can forge a signature for any message x in the framework of HPPK DS scheme.

The overall complexity of the attack is then $\mathcal{O}(p(\frac{S_1}{p} * \frac{S_2}{p}))$. \square

6.2 Forgery Attacks

Proposition 3. *There exists a forgery attack on the HPPK DS scheme with classical computational complexity of $\mathcal{O}(S_1 * S_2)$.*

Proof. To spoof the signature the adversary needs to find values F', H' that satisfy

$$\sum_{k=0}^{\lambda+n} (F' q'_{kj} - s_2 \lfloor \frac{F' \nu_{kj}}{R} \rfloor) x^k \pmod p = \sum_{k=0}^{\lambda+n} (H' p'_{kj} - s_1 \lfloor \frac{H' \mu_{kj}}{R} \rfloor) x^k \pmod p,$$

for all $j \in \{1, \dots, m\}$. This equation can be re-written using the definition of a floor function as

$$\sum_{k=0}^{\lambda+n} (F' q'_{kj} - s_2 \lfloor \frac{F' \nu_{kj}}{R} - a_{kj} \rfloor) x^k \pmod p = \sum_{k=0}^{\lambda+n} (H' p'_{kj} - s_1 \lfloor \frac{H' \mu_{kj}}{R} - b_{kj} \rfloor) x^k \pmod p, \quad (33)$$

for some unknown values $0 \leq a_{kj}, b_{kj} < 1$ for all $j \in \{1, \dots, m\}$. To spoof the signature in the framework of the HPPK DS algorithm, the adversary needs to find values F', H' that satisfy Eq. (33) for all $j \in \{1, \dots, m\}$. This entails that the adversary needs to solve a system of m equation in $2 + m(\lambda + n + 1)$ variables. \square

Proposition 4. *There exists a non-deterministic forgery attack on the HPPK DS scheme with classical computational complexity of $\mathcal{O}(S_1 * S_2)$.*

Proof. For a forgery attack, the adversary wants to find values F' and H' such that

$$\sum_{k=0}^{\lambda+n} \sum_{j=1}^m (F' q'_{kj} - s_2 \lfloor \frac{F' \nu_{kj}}{R} \rfloor) x^k \pmod p = \sum_{k=0}^{\lambda+n} \sum_{j=1}^m (H' p'_{kj} - s_1 \lfloor \frac{H' \mu_{kj}}{R} \rfloor) x^k \pmod p.$$

The adversary also has the ability to intercept honest signatures F, H from communication records that satisfy

$$\sum_{k=0}^{\lambda+n} \sum_{j=1}^m (F q'_{kj} - s_2 \lfloor \frac{F \nu_{kj}}{R} \rfloor) x^k \pmod p = \sum_{k=0}^{\lambda+n} \sum_{j=1}^m (H p'_{kj} - s_1 \lfloor \frac{H \mu_{kj}}{R} \rfloor) x^k \pmod p.$$

It then might be possible to find a positive integer value K such that

$$K \sum_k \sum_j (F q'_{kj} - s_2 \lfloor \frac{F \nu_{kj}}{R} \rfloor) x^k \pmod p = K \sum_k \sum_j (H p'_{kj} - s_1 \lfloor \frac{H \mu_{kj}}{R} \rfloor) x^k \pmod p =$$

$$\begin{aligned} \sum_k \sum_j (KFq'_{kj} - Ks_2 \lfloor \frac{F\nu_{kj}}{R} \rfloor) x^k \pmod p &= \sum_k \sum_j (KHp'_{kj} - Ks_1 \lfloor \frac{H\mu_{kj}}{R} \rfloor) x^k \pmod p = \\ \sum_k \sum_j ((KF)q'_{kj} - s_2 \lfloor \frac{(KF)\nu_{kj}}{R} \rfloor) x^k \pmod p &= \sum_k \sum_j ((KH)p'_{kj} - s_1 \lfloor \frac{(KH)\mu_{kj}}{R} \rfloor) x^k \pmod p = \\ \sum_k \sum_j (F'q'_{kj} - s_2 \lfloor \frac{F'\nu_{kj}}{R} \rfloor) x^k \pmod p &= \sum_k \sum_j (H'p'_{kj} - s_1 \lfloor \frac{H'\mu_{kj}}{R} \rfloor) x^k \pmod p, \end{aligned}$$

where $F' = KF$ and $H' = KH$.

It is important to point out that such value K might not exist for all values F, H . In this case, the adversary would try different honest signature values F, H and accordingly different message value x . The condition for such K to exist for a given F, H is that K multiplied by a decimal part of values $\frac{F\nu_{kj}}{R}$ and $\frac{H\mu_{kj}}{R}$ must be less than 1 for any $k \in \{0, \dots, n + \lambda\}, j \in \{1, \dots, m\}$. Then, $K \lfloor \frac{F\nu_{kj}}{R} \rfloor = \lfloor \frac{KF\nu_{kj}}{R} \rfloor$, and similarly $K \lfloor \frac{H\mu_{kj}}{R} \rfloor = \lfloor \frac{KH\mu_{kj}}{R} \rfloor$.

Since in the worst case scenario the adversary might need to go over all possible F, H , classical computational complexity of this attack is $\mathcal{O}(S_1 * S_2)$. \square

Proposition 5. *There exists a signature forgery attack on HPPK DS scheme with classical complexity of $\mathcal{O}(S_1 * S_2)$.*

Proof. For simplicity, suppose that $k = n + \lambda = 1, m = 1$. However this attack works for any values of n, λ and m . An adversary can intercept an honest signature that satisfies the equation

$$\sum_{k=0}^1 (Fq'_{kj} - s_2 \lfloor \frac{F\nu_{kj}}{R} \rfloor) x^k \pmod p = \sum_{k=0}^1 (Hp'_{kj} - s_1 \lfloor \frac{H\mu_{kj}}{R} \rfloor) x^k \pmod p, \quad (34)$$

for a fixed value $j \in \{1, \dots, m\}$. Given the parameters under the assumption made at the beginning of the proof, Eq. (34) can be expanded as

$$Fq'_{01} - s_2 \lfloor \frac{F\nu_{01}}{R} \rfloor + Fq'_{11}x - s_2 \lfloor \frac{F\nu_{11}}{R} \rfloor x \pmod p = Hp'_{01} - s_1 \lfloor \frac{H\mu_{01}}{R} \rfloor + Hp'_{11}x - s_1 \lfloor \frac{H\mu_{11}}{R} \rfloor x \pmod p.$$

It is true that $(F + tp) \pmod p = F \pmod p$ and $(H + tp) \pmod p = H \pmod p$, for some positive integer t . Under certain conditions, there also exists such positive integer t that

$$\lfloor \frac{F\nu_{kj}}{R} \rfloor \pmod p = \lfloor \frac{(F + tp)\nu_{kj}}{R} \rfloor \pmod p$$

and

$$\lfloor \frac{H\mu_{kj}}{R} \rfloor \pmod p = \lfloor \frac{(H + tp)\mu_{kj}}{R} \rfloor \pmod p.$$

Then signature values $F' = (F + tp)$ and $H' = (H + tp)$ will pass verification in the framework of HPPK DS.

Note however, that the adversary might need to search the whole space of tuples (t, F, H, x) to find at least one to satisfy the verification equation in the framework of HPPK DS scheme. Thus, the classical computational complexity of this attack is $\mathcal{O}(S_1 * S_2)$. \square

6.3 Security Conclusion

In this section, we have comprehensively outlined all the attacks discovered to date on the HPPK DS algorithm. The most potent threats, from the attacker's perspective, manifest as key recovery attacks elucidated in Section 6.1. The classical computational complexities of the attacks detailed in this section are denoted as $\mathcal{O}(2p(S_1 + S_2))$, where s_1 is independent of s_2 in the public key, and $\mathcal{O}(p(\frac{S_1}{p} * \frac{S_2}{p})) = \mathcal{O}(S_1 * S_2/p)$ for s_1 associated with s_2 in the public key—an optimal selection.

The intricacy of these computations hinges on the bit-size of the prime p and the concealed ring values S_1, S_2 . To optimize this complexity, it suffices to choose S_1, S_2 such that $|S_1|_2 = |S_2|_2 = 2|p|_2$, culminating in the total classical computational complexity of the most effective attack on the HPPK DS algorithm being $\mathcal{O}(p^3)$. This, in essence, governs the sizing of the security parameters requisite for each security level.

Table 1 presents a comprehensive overview of key sizes, signature sizes, and estimated entropy for all three NIST security levels. HPPK DS ensures robust security with entropy values of 192, 288, and 384 bits for NIST security levels I, III, and V, respectively.

The public key (PK) sizes exhibit a linear progression, with dimensions of 372, 552, and 704 bytes for security levels I, III, and V. Conversely, the private key (SK) sizes remain remarkably small, comprising 96 bytes for level I, 144 bytes for level III, and 192 bytes for level V.

In terms of signature sizes, HPPK DS maintains efficiency with compact outputs of 128, 192, and 256 bytes for NIST security levels I, III, and V, respectively. Our forthcoming work will delve into the benchmark performance of HPPK DS and provide a thorough comparison with NIST-standardized algorithms.

Table 1: The key and signature sizes in bytes, as provided by the HPPK DS scheme for the proposed parameter sets, are determined based on the optimal complexity of $\mathcal{O}(S_1 S_2 / p)$. In this context, we choose the Barrett parameter R to be 32 bits longer than S_1 / S_2 , and the hidden ring size is set to be $2 \times$ the primed field size. All data is presented in bytes.

Security level	Entropy (bits)	Field size (bits)	PK	SK	$Sign$	$Hash$
I	192	64	372	96	128	SHA-256
III	288	96	552	144	192	SHA-384
V	384	128	704	192	256	SHA-512

7 Conclusion

To counteract the forgery attack observed in MPPK/DS [8, 9], this paper suggests an extension of the HPPK KEM [7] utilizing dual hidden rings for a quantum-secure digital signature scheme. The Barrett reduction algorithm for modular multiplication is expanded to convert modular multiplications over dual hidden rings into divisions by the Barrett parameter R . This embedding process incorporates signature elements into coefficients of a public polynomial over \mathbb{F}_p through the floor function. The non-linear nature of the floor function contributes to the security of the proposed HPPK DS scheme. The functionality of the HPPK DS scheme is illustrated through a toy example. Security analysis reveals that the HPPK DS scheme achieves a complexity of $\mathcal{O}(S_1 S_2 / p) = \mathcal{O}(p^3)$. Future work involves benchmarking its performance and comparing it with NIST-standardized algorithms.

Acknowledgements

The authors extend their sincere gratitude to Michael Redding and Jay Toth for their continuous support and encouragement in tackling the challenges posed by forgery attacks in MPPK DS. Special thanks are also extended to Dr. Brian LaMacchia for engaging discussions throughout the evolution of HPPK DS developments.

Author contributions statement

R.K conceptualized the extension of HPPK KEM for the digital signature scheme, M.P. conducted the security analysis, and M.S. and D.L. delved into the proposal and its security aspects. All authors contributed to and reviewed the manuscript.

References

- [1] Kuang, R. A deterministic polynomial public key algorithm over a prime galois field $\text{gf}(p)$. In 2021 2nd Asia Conference on Computers and Communications (ACCC), 79–88, DOI: 10.1109/ACCC54619.2021.00020 (2021).
- [2] Evdokimov, S. Factorization of polynomials over finite fields in subexponential time under grh. In International Algorithmic Number Theory Symposium, 209–219 (Springer, 1994).
- [3] Kuang, R. and Barbeau, M. Performance analysis of the quantum safe multivariate polynomial public key algorithm. In 2021 IEEE International Conference on Quantum Computing and Engineering (QCE), 351–358 (IEEE, 2021).
- [4] Kuang, R. and Barbeau, M. Indistinguishability and non-deterministic encryption of the quantum safe multivariate polynomial public key cryptographic system. In 2021 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), 1–5 (IEEE, 2021).

- [5] Kuang, R., Perepechaenko, M. and Barbeau, M. A new post-quantum multivariate polynomial public key encapsulation algorithm. *Quantum Inf. Process.* 21, 360 (2022).
- [6] Kuang, R. and Perepechaenko, M. A novel homomorphic polynomial public key encapsulation algorithm [version 1; peer review: awaiting peer review]. *F1000Research* 12, DOI: 10.12688/f1000research.133031.1 (2023).
- [7] Kuang, R. and Perepechaenko, M. Homomorphic polynomial public key encapsulation over two hidden rings for quantum- safe key encapsulation. *Quantum Inf. Process.* 22, 315 (2023).
- [8] Kuang, R., Perepechaenko, M. Barbeau, M. A new quantum-safe multivariate polynomial public key digital signature algorithm. *Sci. Reports* 12, 13168 (2022).
- [9] Kuang, R. and Perepechaenko, M. Optimization of the multivariate polynomial public key for quantum safe digital signature. *Sci. Reports* 13, 6363 (2023).
- [10] Guo, H. An algebraic attack for forging signatures of mppk/ds. *Cryptology ePrint Archive*, Paper 2023/453 (2023).
- [11] Chen, L. et al. Report on post-quantum cryptography, vol. 12 (US Department of Commerce, National Institute of Standards and Technology, 2016).
- [12] Gorjan Alagic and Jacob Alperin-Sheriff and Daniel Apon and David Cooper and Quynh Dang and Yi-Kai Liu and Carl Miller and Dustin Moody and Rene Peralta and Ray Perlner and Angela Robinson and Daniel Smith-Tone. Status report on the first round of the nist post-quantum cryptography standardization process. <https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.8240.pdf> (2019).
- [13] McEliece, R. J. A Public-Key Cryptosystem Based On Algebraic Coding Theory. *Deep. Space Netw. Prog. Rep.* 44, 114–116 (1978).
- [14] Avanzi, R. et al. CRYSTALS-KYBER. Tech. rep. available at <https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions> (2020). National Institute of Standards and Technology.
- [15] Stehle and D. Steinfeld, R. Making ntruencrypt and ntrusign as secure as standard worst-case problems over ideal lattices. *Cryptol. ePrint Arch. Rep.* 2013/004 (2013).
- [16] D’Anvers, J.-P., Karmakar, A., Roy, S. S. Vercauteren, and F. MI wr-based kem. <https://www.esat.kuleuven.be/cosic/pqcrypto/saber/index.html>. Online; accessed 1 November 2023.
- [17] Lyubashevsky, V. et al. CRYSTALS-DILITHIUM. Tech. rep. available at <https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions> (2020). National Institute of Standards and Technology.
- [18] Prest, T. et al. FALCON. Tech. rep. available at <https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions> (2020). National Institute of Standards and Technology.
- [19] Ding, J., Deaton, J., Schmidt, K., Vishakha and Zhang, Z. Cryptanalysis of the lifted unbalanced oil vinegar signature scheme. In *Annual International Cryptology Conference*, 279–298 (Springer, 2020).
- [20] NIST. Status report on the second round of the nist post-quantum cryptography standardization process. <https://csrc.nist.gov/publications/detail/nistir/8309/final> (2021).
- [21] NIST. Status report on the third round of the nist post-quantum cryptography standardization process. <https://csrc.nist.gov/publications/detail/nistir/8413/final> (2022).
- [22] Aumasson, J.-P. et al. SPHINCS+. Tech. rep. available at <https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions> (2020). National Institute of Standards and Technology.
- [23] Jao, D. and De Feo, L. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Yang, B.-Y. (ed.) *Post-Quantum Cryptography*, 19–34 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2011).
- [24] Robert, D. Breaking sidh in polynomial time. *Cryptology ePrint Archive*, Paper 2022/1038 (2022).
- [25] Castryck and W. Decru, T. An efficient key recovery attack on sidh (preliminary version). *Cryptology ePrint Archive*, Paper 2022/975 (2022).
- [26] Wenger, E., Chen, M., Charton, F. and Lauter, K. Salsa: Attacking lattice cryptography with transformers. *Cryptology ePrint Archive*, Paper 2022/935 (2022).
- [27] Li, C. et al. Salsa picante: a machine learning attack on lwe with binary secrets (2023). 2303.04178.
- [28] Li, C. Y., Wenger, E., Allen-Zhu, Z., Charton and F. Lauter, K. Salsa verde: a machine learning attack on learning with errors with sparse small secrets (2023). 2306.11641.
- [29] Sharp, T., Khare, R., Pederson and E. Traversa, F. L. Scaling up prime factorization with self-organizing gates: A memcomputing approach (2023). 2309.08198.
- [30] Zhang, Y.-H. and Ventra, M. D. Implementation of digital memcomputing using standard electronic components (2023). 2309.12437.

-
- [31] Kuang, R. and Perepechaenko, M. Quantum encryption with quantum permutation pad in ibmq systems. EPJ Quantum Technol. 9, DOI: 10.1140/epjqt/s40507-022-00145-y (2022).
 - [32] Perepechaenko, M. and Kuang, R. Quantum encryption of superposition states with quantum permutation pad in ibm quantum computers. EPJ Quantum Technol. 10, DOI: 10.1140/epjqt/s40507-023-00164-3 (2023).
 - [33] Kuang, R. and Barbeau, M. Quantum permutation pad for universal quantum-safe cryptography. Quantum Inf. Process. 21, 211 (2022).
 - [34] Wikipedia. Barrett reduction (2023).