computers

MDPI

Article

Enhancing the Security of Classical Communication with Post-Quantum Authenticated-Encryption Schemes for the Quantum Key Distribution

Farshad Rahimi Ghashghaei [1], Yussuf Ahmed [1], Nebrase Elmrabit [2,3,*] and Mehdi Yousefi [4]

[1] School of Computing and Digital Technology, Birmingham City University, Birmingham B4 7XG, UK
[2] Department of Cyber Security and Networks, Glasgow Caledonian University, Glasgow G4 0BA, UK
[3] College of Computing and Information Technology, Ministry of Technical and Vocational Education, Zawia 218, Libya
[4] Independent Researcher, 12 Riverview Place, Glasgow G5 8EH, UK
* Correspondence: nebrase.elmrabit@gcu.ac.uk

Abstract: This research aims to establish a secure system for key exchange by using post-quantum cryptography (PQC) schemes in the classic channel of quantum key distribution (QKD). Modern cryptography faces significant threats from quantum computers, which can solve classical problems rapidly. PQC schemes address critical security challenges in QKD, particularly in authentication and encryption, to ensure the reliable communication across quantum and classical channels. The other objective of this study is to balance security and communication speed among various PQC algorithms in different security levels, specifically CRYSTALS-Kyber, CRYSTALS-Dilithium, and Falcon, which are finalists in the National Institute of Standards and Technology (NIST) Post-Quantum Cryptography Standardization project. The quantum channel of QKD is simulated with Qiskit, which is a comprehensive and well-supported tool in the field of quantum computing. By providing a detailed analysis of the performance of these three algorithms with Rivest–Shamir–Adleman (RSA), the results will guide companies and organizations in selecting an optimal combination for their QKD systems to achieve a reliable balance between efficiency and security. Our findings demonstrate that the implemented PQC schemes effectively address security challenges posed by quantum computers, while keeping the the performance similar to RSA.

Keywords: post-quantum cryptography; quantum key distribution; NIST; CRYSTALS-Kyber; CRYSTALS-Dilithium; Falcon; Qiskit

Citation: Ghashghaei, F.R.; Ahmed, A.; Elmrabit, N.; Yousefi, M. Enhancing the Security of Classical Communication with Post-Quantum Authenticated-Encryption Schemes for the Quantum Key Distribution. *Computers* **2024**, *13*, 163. https://doi.org/10.3390/computers13070163

Academic Editor: Leandros Maglaras

Received: 10 June 2024
Revised: 23 June 2024
Accepted: 27 June 2024
Published: 1 July 2024
Corrected: 12 February 2025

1. Introduction

In the modern digital age, protecting the confidentiality and integrity of communication data is essential. While modern cryptographic algorithms are effective in classical computing contexts, they are increasingly susceptible to the emerging field of quantum computing. Quantum computers can solve certain problems greatly faster than classical computers. For instance, Shor's algorithm can easily factor large integers and solve the discrete logarithm problem, which places the security of widely used cryptographic algorithms such as Rivest–Shamir–Adleman (RSA) and elliptic curve cryptography (ECC) in risk [1–3]. Similarly, symmetric key algorithms are also challenged by the Grover's algorithm that can speed up brute-force search by reducing their security strength. As these advancements in quantum technologies continue to progress, the demand for innovative cryptographic solutions is becoming increasingly urgent. Post-quantum cryptography (PQC) is a developing area that provides cryptographic algorithms that are resistant to quantum attacks [4]. Simultaneously, quantum key distribution (QKD) protocols are being developed to utilize the principles of quantum mechanics to facilitate secure key exchange, thereby achieving unconditional security. Quantum computers pose a great challenge to

traditional cryptographic methods, as noted in [5]. However, PQC algorithms, including lattice-based cryptography, code-based cryptography , hash-based cryptography, and multivariate cryptography [6–14], provide strong protection against the computational abilities of quantum adversaries. These adversaries can easily solve classical hard problems like integer factorization and discrete logarithms. It is important to highlight an advantage of PQC encryption algorithms, which is their built-in resilience against attacks from quantum computers as emphasized by [15]. In the complex realm of QKD, security is deeply rooted in the principles of quantum mechanics such as Heisenberg's uncertainty principle and the no-cloning theorem, explained by [16,17] respectively. These principles act as guardians, promptly detecting any attempts at interception and strengthening the security of distributed keys. The importance of PQC in ensuring the security of communication channels is underscored by [18], given the reliance of QKD on the distribution of secret keys between parties. Moreover, the vulnerabilities revealed in modern cryptographic methods, particularly when faced with quantum threats [19], show the meaning of adopting PQC methodologies. By enhancing authentication and encryption processes, PQC schemes offer an improved approach for integrating new participants into QKD, enhancing accessibility while reinforcing secure communication.

The crucial role of PQC in QKD systems is further emphasized by diverse experimental efforts, similar to those suggested by [4]. This research aims to establish a secure system for key exchange by using PQC schemes in the classic channel of QKD. Other objectives include finding the optimal balance between security and communication speed among different security levels of CRYSTALS-Kyber, CRYSTALS-Dilithium, and Falcon, which are finalists in the National Institute of Standards and Technology (NIST) Post-Quantum Cryptography Standardization project. The quantum channel of QKD is simulated with Qiskit, a comprehensive and well-supported tool in the field of quantum computing. The results will guide organizations in selecting optimal security level for their QKD systems to achieve a reliable balance between efficiency and security. We will achieve this by providing a detailed analysis of the performance of these three algorithms with RSA. Our findings will show that the implemented PQC schemes effectively improve the reliability of communication by addressing security challenges and having the RSA performance.

This study makes several significant contributions. First, it introduces an innovative approach to enhancing the security of the classical channel by combining PQC with QKD, thereby increasing overall safety. Second, it provides an in-depth assessment of various algorithm parameters, including sizes, speeds, and security levels. This comparative analysis aims to identify the optimal combination for the proposed cryptosystem. Third, it explores the practical implementation challenges of integrating PQC into a QKD system. Finally, it provides guidelines for future research and development and shows key areas for further exploration to enhance the robustness and efficiency of quantum-secure communication systems.

The structure of this paper is organized as follows: First, we review relevant studies on the QKD BB84 protocol and the CRYSTALS-Kyber, CRYSTALS-Dilithium, and Falcon algorithms. Next, we provide a detailed explanation of our methodology, covering the requirements and experimental simulations. Following this, we present our experimental results, compare them with RSA performance, and discuss their significance. We also examine the potential drawbacks of the proposed method. Finally, we conclude the paper with a discussion of possible directions for future research.

2. Literature Review

The BB84 protocol is a QKD protocol that was proposed by Charles Bennett and Gilles Brassard in 1984 [20]. It is one of the most widely used QKD protocols and is named after the surnames of its inventors and the year of its invention. CRYSTALS-Kyber is another technique used to safeguard messages, especially against newer, more powerful computers that could break traditional encryption methods [21]. It relies on advanced mathematical concepts to create codes that are very difficult for these computers to crack.

At its core, lattice-based cryptography takes on the complexities of mathematical problems built within lattice structures. These complex problems form the basis upon which the security of lattice-based schemes is established, effectively protecting them from potential attacks from quantum computers [22]. These computers act as an approaching threat to modern digital signature mechanisms as well. CRYSTALS-Dilithium appears as a signature resistant to these quantum attacks and is secure against side-channel attacks as well [23]. As digital communication evolves, a strong and safe method of creating digital signatures is provided by this algorithm, which is at the cutting edge of the evolution at this moment, where the need for digital signatures that are resistant to quantum incidents is urgent. Falcon also appears to be an appropriate choice for safe digital signatures in the rapidly developing area of post-quantum cryptography. With its foundation based on lattice-based cryptography, Falcon addresses an important need for cryptographic algorithms that maintain effectiveness and practicality while dealing with the power of quantum computers [24,25]. The strength of all three PQC algorithms comes from the complexity of lattice problems, which is a mathematical idea that serves as the core of their security.

2.1. Quantum Key Distribution BB84 Protocol

The BB84 protocol is designed to ensure unconditional security in the transmission of a shared secret key between two parties, Alice and Bob [26]. The BB84 protocol uses quantum bits (qubits) for the transmission of data in the quantum channel. The process involves encoding data bits into various polarized states of photons, creating qubits essential for secure key transmission within the quantum channel [27]. In quantum communication, polarized states often appear as either vertical and right diagonal (traditionally denoted as "1") or horizontal and left diagonal (usually indicated as "0"). Table 1 shows the use of two distinct bases, labeled as "+" and "×", for detecting these photons [28].

Table 1. Security of BB84 protocol quantum channel.

Qubit	Alice's Basis	Alice's Bit	Eve's Basis	Eve's Bit	Bob's Basis	Bob's Bit	Basis Match	Bit Match	Error
$\|\psi\rangle_1$	+	0	×	0	×	1	No	-	No
$\|\psi\rangle_2$	×	1	+	1	+	1	No	-	No
$\|\psi\rangle_3$	+	0	+	0	+	0	Yes	Yes	No
$\|\psi\rangle_4$	×	1	×	1	×	1	Yes	Yes	No
$\|\psi\rangle_5$	+	0	+	0	+	1	Yes	No	Yes
$\|\psi\rangle_6$	×	1	+	1	+	0	No	-	No
$\|\psi\rangle_7$	+	0	×	0	×	0	No	-	No
$\|\psi\rangle_8$	×	1	×	0	×	0	Yes	No	Yes

The quantum state $|\psi\rangle$ is represented as a linear combination of basis states [29]:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \quad (1)$$

where α and β are complex probability amplitudes.

Basis states in the rectilinear basis (Z basis) and diagonal basis (X basis):

$$|0\rangle : \text{Bit value } 0$$
$$|1\rangle : \text{Bit value } 1$$

$$|+\rangle : \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) : \text{Bit value } 0$$
$$|-\rangle : \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) : \text{Bit value } 1$$

Classical post-processing in BB84 protocol:

As can be seen in Table 2, the BB84 protocol involves four main steps: key sifting, error correction, key reconciliation, and privacy amplification [30]. Firstly, in key sifting, Bob sends the bases he used to Alice to confirm against the qubits she sent. Then, during error correction, Alice tells Bob the correct bases, and Bob discards any bits measured incorrectly. After this, the "quantum bit error rate (QBER)" is calculated. This rate is found by comparing the results of some qubits' measurements between Alice and Bob to see how often they disagreed [31]. If there has been interference from an eavesdropper, called Eve, the QBER goes over a certain level. In that case, both Alice and Bob throw out the keys and start over. Even if the QBER is below the threshold, it is still not zero, meaning that Alice and Bob do not have the same keys. Therefore, in addition to correcting errors, they use the cascade protocol for key reconciliation. According to this protocol, Alice and Bob segment the bits into fixed-size blocks, compute the parity of each block, and exchange these data for error correction. The BB84 protocol concludes with privacy amplification, where the shared secret key undergoes further processing to bolster its security. This involves applying a one-way function that reduces the information accessible to potential eavesdroppers. Subsequently, the final key is utilized for encrypting and decrypting messages exchanged between Alice and Bob.

Table 2. Security of BB84 protocol classic channel.

Step	Description
1. Key Sifting	Bob's sent bases
2. Error Correction	Correct basis exchange
2.1. QBER Calculation	Calculate QBER
2.2. Threshold Check	Check QBER threshold
3. Key Reconciliation	Correct errors
4. Privacy Amplification	Enhance key security

The BB84 protocol has proven its resilience against diverse attacks, including man-in-the-middle and eavesdropping attempts [32]. This robustness arises from any effort to intercept or measure the qubits, inevitably altering their polarization, thus alerting the receiver. Consequently, any eavesdropping attempt is promptly detected, preventing the establishment of the shared secret key. Nevertheless, the BB84 protocol is not without limitations. A primary drawback is its reliance on a costly and challenging-to-implement quantum channel. Also, it requires a classical channel for error correction and communication which is susceptible to potential attacks [33]. To avoid additional overheads like key reconciliation and privacy amplification in the classical channel, there are some authenticated encryption schemes which can be used in the classical channel using the PQC algorithms introduced by NIST [34].

2.2. CRYSTALS-Kyber

CRYSTALS-Kyber is an advanced lattice-based key exchange protocol designed to ensure the integrity of transmitted information. At its core, CRYSTALS-Kyber leverages the module learning with errors (M-LWE) technique, an evolution of the original learning with errors (LWE) problem [35]. This technique enhances computational efficiency while maintaining cryptographic robustness, making CRYSTALS-Kyber a reliable solution for secure key exchange in real-world scenarios [36]. The M-LWE technique is central to the efficacy of CRYSTALS-Kyber. It optimizes the LWE problem, balancing computational efficiency and cryptographic strength [37]. This optimization ensures the practicality of lattice-based cryptography in real-world applications [36]. CRYSTALS-Kyber moves beyond theoretical constructs by providing a key encapsulation mechanism (KEM). This mechanism enhances the protocol's versatility and utility by enabling secure key establishment through the encapsulation of a symmetric key with a public key. This encapsulated key can be securely exchanged between parties, forming the foundation for robust communication encryption [36].

As described in [38] work, a double-NTRU (D-NTRU)-based KEM with IND-CCA2 security highlights the importance of parameter considerations. The approval of CRYSTALS-Kyber, an NIST Post-Quantum Cryptography finalist with security levels Kyber-512, Kyber-768, and Kyber-1024, complies with this emphasis on parameters [34,35]. The specified parameter sets for Kyber, detailed in Table 3, determine values for n, k, q, η, d_u, d_v (control compression of $(\mathbf{u}, v)$), and δ (the chance of decryption producing an error), ensuring diverse levels of security and efficiency.

Table 3. Parameter sets for CRYSTALS-Kyber.

Security Level	n	k	q	η	d_u	d_v	δ
Kyber 512	256	2	3329	2	10	4	2^{-139}
Kyber 768	256	3	3329	2	10	4	2^{-164}
Kyber 1024	256	4	3329	2	11	5	2^{-174}

Using CRYSTALS-Kyber for secure key exchange, especially for AES-256 encryption [39], is sensible because of the rising vulnerability of lower AES levels to new algorithms like Grover, which represent a risk. This key is for AES 256-bit encryption which ensures protection against emerging cryptographic threats, including those from quantum advancements. Algorithms 1–3 provide an overview of the algorithmic structure of the CRYSTALS-Kyber scheme.

Key generation: The Kyber key generation process commences with the generation of random seeds ρ and σ. Subsequently, a public matrix $\mathbf{A}$ is sampled from a ring with dimension $k \times k$. Random vectors $\mathbf{s}$ and $\mathbf{e}$ are then drawn from error distributions, contributing to the randomness and security of the key pair. The compressed vector $\mathbf{t}$ is computed by compressing the matrix-vector product of $\mathbf{A}$ and $\mathbf{s}$ added to $\mathbf{e}$, encapsulating essential information about the key pair. The public key pk is meticulously formed by combining the generated seeds and the compressed vector, while the secret key sk is simply the vector $\mathbf{s}$, thereby completing the key generation process in a manner that ensures both the privacy and integrity of the cryptographic system.

Algorithm 1 Crystal-Kyber key generation [35].

$\rho, \sigma \leftarrow \{0,1\}^{256}$
$\mathbf{A} \sim R_q^{k \times k} := \mathrm{Sam}(\rho)$
$(\mathbf{s}, \mathbf{e}) \sim \beta_\eta^k \times \beta_\eta^k := \mathrm{Sam}(\sigma)$
$\mathbf{t} := \mathrm{Compress}_q(\mathbf{As} + \mathbf{e})$
return $(pk := (\rho, \mathbf{t}), sk := \mathbf{s})$

CCAKEM encryption: Kyber CCAKEM encryption takes a message m and generates a shared key K along with a ciphertext c. The process involves generating a random key $\hat{K}$ and a nonce r. The Kyber CPA encryption algorithm is then applied to generate the ciphertext $(\mathbf{u}, v)$ using the public key and message. The shared key K is derived from hashing $\hat{K}$ and the hash of the ciphertext.

Algorithm 2 Kyber.CCAKEM.Enc(pk) [35].

$m \leftarrow \{0,1\}^{256}$
$(\hat{K}, r) := \mathrm{G}(\mathrm{H}(pk), m)$
$(\mathbf{u}, v) := \mathrm{Kyber.CPA.Enc}((\mathbf{t}, \rho), m; r)$
$c := (\mathbf{u}, v)$
$K := \mathrm{H}(\hat{K}, \mathrm{H}(c))$
return (c, K)

Key exchange verification: In the Kyber key exchange verification process, upon receiving a ciphertext $(\mathbf{u}', v')$, the decryption takes place using the secret key $\mathbf{s}$. Subsequently, the shared key $\hat{K}'$ and nonce r' are recalculated in a manner analogous to the encryption procedure. To ensure the integrity of the received ciphertext, the Kyber CPA encryption algorithm is once again applied, regenerating a new ciphertext $(\mathbf{u}', v')$. The comparison between the regenerated and received ciphertexts is pivotal: if they match, the shared key K is computed by hashing $\hat{K}'$ and the hash of the received ciphertext; conversely, if a mismatch occurs, an alternative key is derived by hashing a predetermined placeholder value z along with the hash of the received ciphertext. This multilayered verification mechanism solidifies the security and reliability of the Kyber key exchange protocol.

Algorithm 3 Crystal-Kyber key exchange verification [35].

Require: $m' := \text{Kyber.CPA.Dec}(\mathbf{s}, (\mathbf{u}, v))$
  $(\hat{K}', r') := \text{G}(\text{H}(pk), m')$
  $(\mathbf{u}', v') := \text{Kyber.CPA.Enc}((\mathbf{t}, \rho), m'; r')$
  if $(\mathbf{u}', v') = (\mathbf{u}, v)$ **then**
    return $K := \text{H}(\hat{K}', \text{H}(c))$
  else
    return $K := \text{H}(z, \text{H}(c))$
  end if

2.3. CRYSTALS-Dilithium

The CRYSTALS-Dilithium scheme, a robust lattice-based digital signature scheme, stands as a stalwart guardian against the potential threats posed by quantum computers. In the evolving landscape of post-quantum cryptography, CRYSTALS-Dilithium shines as a beacon of security, ensuring the integrity and authenticity of digital signatures, even in the face of quantum advancements [40]. The practicality of CRYSTALS-Dilithium extends beyond theoretical constructs, ensuring that its security benefits are not just conceptual but also accessible and applicable in real-world scenarios [41]. Rigorous analysis and extensive proofs confirm the strong security of this scheme, reflecting the careful work of cryptographic researchers [42]. The design of CRYSTALS-Dilithium meets the highest standards of verifiable security, showing its trustworthiness and suitability for protecting sensitive digital transactions and communications. Notably, CRYSTALS-Dilithium has earned its prominence by being selected as a finalist in the NIST Post-Quantum Cryptography Standardization process [34]. This recognition highlights the scheme's significance in the global cryptographic community and its potential to shape the future of secure digital communication. With its lattice-based architecture, flexibility in design, and practical applicability, CRYSTALS-Dilithium charts a strategic course in advancing secure digital signatures, effectively enhancing them against the looming quantum challenges anticipated with the widespread adoption of QKD in our networks.

CRYSTALS-Dilithium stands as an NIST finalist that offers various security levels designed for different requirements [34]. The scheme offers security levels based on NIST standards, including levels 2, 3, and 5. Each security level corresponds to specific parameter sets, ensuring a balance between security and computational efficiency as can be seen in Table 4.

Algorithms 4–6 provide an excellent foundation for the crucial CRYSTALS-Dilithium scheme in the context of the article that was submitted to NIST. These algorithms serve as the foundation for the main methods of cryptography and play a crucial part in ensuring the security of communication.

Table 4. Parameter sets for CRYSTALS-Dilithium.

Parameters	2	3	5
q	8,380,417	8,380,417	8,380,417
γ_1	2	17	19
γ_2	$(q-1)/88$	$(q-1)/32$	$(q-1)/32$
(k, ℓ)	(4, 4)	(6, 5)	(8, 7)
η	2	4	2
β	78	196	120
ω	80	55	75

Key generation: Key generation is responsible for generating the cryptographic keys which create the secret key (sk) for signature generation and the public key (pk) for signature verification. This process begins with the utilization of seeds ρ and ρ', alongside a key, to expand a public matrix A using the AES algorithm. This matrix A is structured as a $(k \times \ell)$ matrix and is composed of polynomials within the ring R_q and $\zeta = 256$. The seeds ρ' and a nonce are employed to generate vectors s_1 and s_2, with s_1 being of size l and s_2 being of size k. The multiplication of matrix A and vector s_1 is achieved through the forward number theoretic transform (NTT). The process iterates for the size of s_2 (or k times), wherein each iteration involves the multiplication of a single row of A and s_1, with the result stored in t. The matrix multiplication concludes by adding s_2 to t, followed by the reduction of the coefficients of t. Subsequently, t_0 and t_1 are separated from t, and a combination of ρ and t_1 is utilized to form pk. The function shake256 generates an output tr based on the input pk. The formation of sk involves the amalgamation of ρ, the key, tr, s_1, s_2, and t_0. Notably, the key generation process yields both sk and pk at the same time.

Algorithm 4 Dilithium key generation [43].

Require: $\zeta \in \{0,1\}^{256}$
Ensure: $(\rho, \rho', K) \in \{0,1\}^{256} \times \{0,1\}^{512} \times \{0,1\}^{256} := H(\zeta)$
$A \in \mathbb{R}^{k \times \ell} := \text{ExpandA}(\rho)$
$(s_1, s_2) \in S^{\ell} \times S^{k} := \text{ExpandS}(\rho')$
$t := As_1 + s_2$
$(t_1, t_0) := \text{Power2Round}_q(t, d)$
$tr \in \{0,1\}^{256} := H(\rho \| t_1)$
return $(pk = (\rho, t_1), sk = (\rho, K, tr, s_1, s_2, t_0))$

Signature generation: The cryptographic signature generation process begins by extracting seeds and values from sk, laying the foundation for crafting a robust signature. These elements, extracted with precision, play a pivotal role in shaping the ensuing signature. Alongside this extraction, the input message seamlessly integrates into the signature creation process, ensuring that the resulting signature faithfully represents the original message. In tandem with this integration, a collision-resistant hash function, μ, undergoes computation, leveraging the message and additional inputs to fortify the integrity of the signature generation process. This hash serves as a critical component, adding an extra layer of security to the cryptographic framework. Furthermore, the expansion of matrix A marks a significant step in enhancing the security posture of the algorithm. The subsequent application of forward number theoretic transform (NTT) to pertinent vectors reinforces the cryptographic resilience of the system, contributing to the overall robustness of the signature generation process. As the execution unfolds, an infinite loop orchestrates the generation of an intermediate vector y, which undergoes meticulous scrutiny through matrix multiplication and various validation functions. This iterative process

ensures that the signature meets stringent criteria, affirming its validity and reliability in cryptographic applications.

Algorithm 5 Dilithium signature generation [43].

Require: $sk = (\rho, K, tr, s_1, s_2, t_0), M \in \{0,1\}^*$
Ensure: $\sigma = (\hat{c}, z, h)$
$A \leftarrow \text{ExpandA}(\rho) \quad \mu \leftarrow H(tr \| M) \quad \rho' \leftarrow H(K \| \mu)$
$k \leftarrow 0 \quad$ abort $\leftarrow 1$
while abort **do**
abort $\leftarrow 0$
$y \leftarrow \text{ExpandMask}(\rho', k)$
$w \leftarrow A \cdot y$
$w_1 \leftarrow \text{HighBits}$
$\hat{c} \leftarrow H(\mu \| \omega_1)$
$c \leftarrow \text{SampleInBall}(\hat{c})$
$z \leftarrow y + c \cdot \gamma_1$
$r_0 \leftarrow \text{LowBits}(\omega - \text{c} \cdot s_2, 2\gamma_2)$
if $\|z\|_\infty \geq \gamma_1 - \beta$ or $\|r_0\|_\infty \geq \gamma_2 - \beta$ **then**
abort $\leftarrow 1$
else
$h \leftarrow \text{MakeHint}(-c \cdot t_0, \omega - c \cdot s_2 + c \cdot t_0, 2\gamma_2)$
if $\|c \cdot t_0\|_\infty \geq \gamma_2$ or $\sum h_i > \omega$ **then**
abort $\leftarrow 1$
end if
$k \leftarrow k + \ell$
end if
end whilereturn $\sigma = (\hat{c}, z, h)$

Signature verification: The algorithm verifies the authenticity of a signature against its corresponding *pk*, ensuring secure communication. It begins by extracting the signature and *pk* components. If the signature meets predefined conditions, indicating authenticity, the algorithm accepts it and copies the message. Otherwise, it promptly rejects the signature, safeguarding against potential tampering or unauthorized access.

Algorithm 6 Dilithium signature verification [43].

Require: $pk = (\rho, t_1), M \in \{0,1\}^*, \sigma = (\hat{c}, z, h)$
Ensure: Valid or Invalid
A $\leftarrow \text{ExpandA}(\rho)$
$\mu \leftarrow H(H(\rho \| t_1) \| M)$
$c \leftarrow \text{SampleInBall}(\hat{c})$
$(\omega_1, \omega_0) \leftarrow \text{UseHint}(h, A \cdot z - c \cdot t_1 \cdot 2^d)$
if $\|z\|_\infty < \gamma_1 - \beta$ and $\hat{c} = H(\mu \| \omega_1)$ and $\sum h_i \leq \omega$ **then**
return Valid
return Invalid

2.4. Falcon

Falcon, with its foundation based on lattice-based cryptography, addresses an important need for cryptographic algorithms that maintains effectiveness and practicality while dealing with the power of quantum computers [24,25]. One of the outstanding features of this scheme is its remarkable efficiency in both signature generation and verification processes [44]. This efficiency sets Falcon apart by providing a solution that is not only secure against quantum adversaries but also practical for real-world deployment. The ability to achieve robust security without sacrificing performance positions Falcon as a vital candidate in the post-quantum cryptographic landscape. Its selection as another

finalist in the NIST Post-Quantum Cryptography Standardization process underscores its credibility and the community's confidence in its accuracy in cryptography [34]. Being an NIST finalist demonstrates its potential to become a recommended standard for secure communication in the future. Beyond its theoretical prowess, however, some researchers have reported problems with configuration and key generation timing, which can be a block to its widespread adoption [45]. Despite these challenges, its small public and private key sizes make it an attractive option for many applications, and ongoing research and development efforts are aimed at addressing these configuration issues. Although Falcon is undoubtedly a great choice in post-quantum cryptography, it is essential to note that NIST has selected CRYSTALS-Dilithium as its first choice for PQC digital signature schemes [34]. While Falcon has earned its place as an NIST finalist in standardization process and has received significant attention and interest within the cryptographic community, NIST's preference for CRYSTALS-Dilithium shows the thorough assessment and selection process used to identify the best options for ensuring our digital future in a post-quantum era.

Falcon features parameter sets tailored to different security levels, providing flexibility in selecting an appropriate configuration based on specific requirements. The parameters, summarized in Table 5, play a crucial role in shaping Falcon's security and efficiency. The parameter sets include values for n (dimension), ϕ (modulus polynomial), q (modulus), β^2 (security parameter), signature size, and public key size. These parameters allow users to customize Falcon's configuration to achieve the desired balance between security and performance.

Table 5. Parameter sets for Falcon.

Parameters	Falcon 256	Falcon 512	Falcon 1024
n	256	512	1024
ϕ	x^{n+1}	x^{n+1}	x^{n+1}
q	12,289	12,289	12,289
β	16,468,416	43,533,782	87,067,565

During its setup process, Falcon employs a unique mathematical polynomial and a numerical input, employing these elements in a manner outlined in Algorithms 7–9. These algorithms work together to produce both the confidential secret key and the publicly accessible key, establishing the cryptographic foundation upon which the Falcon system operates securely and effectively.

Algorithm 7 Falcon key generation KeyGen(ϕ, q) [46].

Require: A monic polynomial $\phi \in \mathbb{Z}[x]$, a modulus q
Ensure: A secret key (sk), a public key (pk)
$\mathbf{f}, \mathbf{g}, \mathbf{F}, \mathbf{G}, \mathbf{fl} \leftarrow \text{NTRUGen}(\phi, q)$
$\mathbf{B} \leftarrow \begin{bmatrix} g & -f \\ G & -F \end{bmatrix}$
$\hat{\mathbf{B}} \leftarrow \text{FFT}(\mathbf{B})$
$\mathbf{G} \leftarrow \hat{\mathbf{B}} \times \hat{\mathbf{B}}^*$
$\text{T} \leftarrow \text{ffLDL}(\mathbf{G})$
for each leaf of T **do**
  leaf.value $\leftarrow \frac{\sigma}{\sqrt{\text{leaf.value}}}$
end for
$\text{sk} \leftarrow (\hat{\mathbf{B}}, \text{T})$
$h \leftarrow gf^{-1} \bmod q$
$\text{pk} \leftarrow h$
return sk, pk

Key generation: Algorithm 7 outlines the process of generating FALCON key pairs, which involves computing polynomials f, g, F, G, and h based on specific equations. These equations define relationships between the polynomials, ensuring the integrity of the generated keys.

To calculate these polynomials, a random number is generated which serves as a seed for initializing shake256. By using shake256 random numbers, the algorithm generates random polynomials f and g with a Gaussian distribution. If the squared norm of these polynomials exceeds predefined bounds or if the orthogonalized vector norms deviate from expected values, new polynomials are generated. The orthogonalized vector norm computation employs the fast Fourier transform (FFT) for efficiency.

The equations guiding the polynomial computation are as follows:

$$fG - gF = q \mod \phi \tag{2}$$

$$h = gf^{-1} \mod (\phi, q) \tag{3}$$

Having obtained the f and g polynomials, the algorithm proceeds to compute the public key polynomial h, fulfilling the requirements of the first equation. Additionally, it solves the second equation (NTRU equation) to derive polynomials F and G. For the sk, the algorithm sequentially encodes the f, g, F, and G polynomials. Meanwhile, pk is encoded by representing the polynomial h. Ultimately, the algorithm generates both the sk and pk, ensuring the cryptographic integrity of the Falcon scheme.

Signature generation: Algorithm 8 outlines the steps for generating signatures in the FALCON scheme. It begins by generating a random seed for the hash function, followed by initializing the shake256 function with this seed. Subsequently, the algorithm computes a hash digest c from the salt r and the input message m. Next, sk, previously encoded during key generation, is decoded to retrieve the polynomials f, g, F, and G. If G is not extracted from sk, the algorithm calculates it. Leveraging these polynomials, the algorithm computes two short vectors s_1 and s_2 satisfying $s_1 \equiv s_2 h \mod q$, all while keeping sk hidden. The short vector s_2 is then encoded and concatenated with the signature length, salt, message, and encoded s_2. These concatenated data are stored in the signature (sig). This process ensures the secure generation of FALCON signatures while maintaining the confidentiality of sk.

Algorithm 8 Falcon signature generation [46].

Require: A message m, a secret key sk, a bound $\lfloor \beta^2 \rfloor$
Ensure: A signature sig of m
r $\leftarrow \{0,1\}^{320}$ uniformly
$c \leftarrow$ HashToPoint(r‖m)
$\mathbf{t} \leftarrow \left(-\frac{1}{q}\text{FFT}(c) \odot \text{FFT}(F), \frac{1}{q}\text{FFT}(c) \odot \text{FFT}(f)\right)$
do
  do
    z $\leftarrow$ ffSampling(t, T)
    $\text{S} = (\text{t} - \text{z})\begin{bmatrix} \frac{FFT(g)}{FFT(G)} & \frac{-FFT(f)}{FFT(F)} \end{bmatrix}$
  while $\|\mathbf{S}\|^2 > \beta^2$
  $(s_1, s_2) \leftarrow$ invFFT(**s**)
  $s \leftarrow \text{Compress}(s_2, 8 \cdot \text{sbytelen} - 328)$
while $(s = \perp)$
return sig = (r, s)

Signature verification: The verification algorithm, described in Algorithm 9, initiates by computing the combination of the initial component of the signature and the message using the HashToPoint function, resulting in a point on a modulo q. This step lays the groundwork for further validation. Following this, the algorithm proceeds to decompress

the second component of the signature. If the decomposition fails, indicating potential tampering or invalidity, the algorithm promptly rejects the signature, ensuring the integrity of the verification process. Subsequently, the algorithm computes s_1 by subtracting the hash digest c from the decompressed s_1 multiplied by pk, all under modulo q. This computation is crucial in validating the authenticity of the signature against the provided pk. A pivotal aspect of the verification process is the evaluation of the squared aggregate vector (s_1, s_2) against a predefined bound $\lfloor \beta^2 \rfloor$. If the squared norm of this vector meets or falls below the specified threshold, indicating adherence to expected parameters, the algorithm accepts the signature. However, if the squared norm exceeds the bound, signifying potential irregularities or deviations from expected behavior, the algorithm rejects the signature, safeguarding against potential security threats or inaccuracies in the verification process.

Algorithm 9 Falcon verification $(\mathrm{m}, \mathrm{sig}, \mathrm{pk}, \lfloor \beta^2 \rfloor)$ [46].

Require: A message m, a signature $\mathrm{sig} = (\mathrm{r}, \mathrm{s})$, a public key $\mathrm{pk} = h \in \mathbb{Z}_q[x]/(\phi)$, a bound $\lfloor \beta^2 \rfloor$
Ensure: Accept or Reject
$c \leftarrow \mathrm{HashToPoint}(\mathrm{r} \| \mathrm{m}, q, n)$
$s_2 \leftarrow \mathrm{Decompress}(\mathrm{s}, 8 \cdot \mathrm{sbytelen} - 328)$
if $s_2 = \perp$
  return Reject
$s_1 \leftarrow c - s_2 h \bmod q$
if $\|(s_1, s_2)\|^2 \leq \lfloor \beta^2 \rfloor$
  return Accept
else
  return Reject

2.5. Evaluation of PQC Algorithms

Before evaluating the three algorithms, it is essential to explain lattice-based cryptography, as they all operate using this method. Lattice-based cryptography relies on the hardness of lattice problems which are believed to be resistant to quantum attacks. Although some researchers have suggested that lattice-based cryptography might be vulnerable because of potential algorithmic weaknesses, further studies have identified errors in these assessments which confirmed the security of lattice-based cryptography methods again [47]. CRYSTALS-Kyber is efficient and secure due to the hardness of module lattice problems, with relatively small key sizes. However, it has potential side-channel vulnerabilities and larger ciphertext sizes compared to traditional algorithms. CRYSTALS-Dilithium offers fast signing and verification with robust security, but has larger public keys and signatures, and its implementation can be complex for some applications. Falcon features compact signatures and high verification speed, supported by a strong theoretical foundation. However, it can be complex to implement, may face numerical stability issues, and has larger key sizes. While previous research on using PQC schemes for QKD introduced valuable insights [48–50], there is still a research gap regarding the optimal solution for addressing both encryption and authentication in QKD. In the next section, we introduce our approach to answer this question.

3. Methodology

This methodology outlines our comprehensive approach to exploring the integration of quantum key distribution and post-quantum cryptography, with a primary focus on leveraging Qiskit as a quantum simulation platform. Qiskit, an open-source software development kit (SDK) developed by IBM Research, serves as an effective research tool, enabling us to investigate quantum communications and its advantages over traditional cryptographic primitives [51].

3.1. Quantum Simulation with Qiskit

The methodology begins by using Qiskit for modeling quantum processes. Within this advanced quantum simulation environment, we explored the complexities of quantum computations, allowing us to model and evaluate potential quantum attacks. This advanced quantum computing framework is critical to ensuring the practicability and high precision of our research. It makes it easier to generate simulated data, which provide useful insights into quantum system operation and possible vulnerabilities. These simulated data become an important component for testing and confirming the strength of classical communication channels against quantum threats, hence contributing to a thorough understanding of quantum-resistant cryptographic algorithms and their real-world applications.

3.2. BB84 Protocol

In our experimental methodology, we proceed with the understanding that the security of the BB84 protocol has been carefully tested and verified in prior research papers [32,52,53]. As a result, we focus on the practical implementation and performance aspects of BB84 without directly considering the presence of an eavesdropper (Eve) in our specific experiments. Our objective is to assess the applicability and efficiency of the protocol within the confines of a controlled and without any eavesdropper. Since QKD works with the no-cloning theorem, which states that it is impossible to create perfect copies of an unknown quantum state [17], it will be impossible for Eve to make a perfect copy of a transmitting qubit over the quantum channel. Any act on the qubit will ultimately cause some errors, which can be detected by both Alice and Bob.

As previously discussed in the literature review, consider a quantum state $|\psi\rangle$ expressed as a linear combination of basis states:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{4}$$

Suppose there exists a unitary operator U that can clone any quantum state:

$$U(|\psi\rangle \otimes |0\rangle) = |\psi\rangle \otimes |\psi\rangle \tag{5}$$

To test this assumption, substitute $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ into the cloning operation:

$$U((\alpha|0\rangle + \beta|1\rangle) \otimes |0\rangle) = (\alpha|0\rangle + \beta|1\rangle) \otimes (\alpha|0\rangle + \beta|1\rangle) \tag{6}$$

Expanding the right-hand side:

$$(\alpha|0\rangle + \beta|1\rangle) \otimes (\alpha|0\rangle + \beta|1\rangle) = \alpha^2|0\rangle \otimes |0\rangle + \alpha\beta|0\rangle \otimes |1\rangle + \beta\alpha|1\rangle \otimes |0\rangle + \beta^2|1\rangle \otimes |1\rangle \tag{7}$$

Next, considering the linearity property of unitary operators and assuming perfect cloning by U:

$$U((\alpha|0\rangle + \beta|1\rangle) \otimes |0\rangle) = \alpha U(|0\rangle \otimes |0\rangle) + \beta U(|1\rangle \otimes |0\rangle) \tag{8}$$

Given U performs perfect cloning:

$$U(|0\rangle \otimes |0\rangle) = |0\rangle \otimes |0\rangle, \quad U(|1\rangle \otimes |0\rangle) = |1\rangle \otimes |1\rangle \tag{9}$$

Substituting these results:

$$\alpha U(|0\rangle \otimes |0\rangle) + \beta U(|1\rangle \otimes |0\rangle) = \alpha(|0\rangle \otimes |0\rangle) + \beta(|1\rangle \otimes |1\rangle) \tag{10}$$

Comparison of this with the expanded form reveals:

$$\alpha(|0\rangle \otimes |0\rangle) + \beta(|1\rangle \otimes |1\rangle) \neq \alpha^2|0\rangle \otimes |0\rangle + \alpha\beta|0\rangle \otimes |1\rangle + \beta\alpha|1\rangle \otimes |0\rangle + \beta^2|1\rangle \otimes |1\rangle \tag{11}$$

This difference shows that no unitary operator U can clone an arbitrary quantum state. As a consequence, the QKD quatnum channel will be considered secure against eavesdropping attacks.

The QKD protocol involves a series of steps to secure quantum communication:

1. **Alice's initialization**: Alice begins the quantum key distribution process by choosing a set of random bits and corresponding random bases. She keeps this information private to avoid unauthorized access.
2. **Encoding qubits**: Alice encodes each bit into a string of qubits using the selected bases. This encoding process is necessary to ensure trustworthy transmission. She sends the encrypted data to Bob in the form of a qubit string, which is Alice's result.
3. **Bob's measurement**: When Bob receives Alice's encoded qubit string, he randomly measures each qubit using his own set of randomly selected bases. By keeping the measurement results private, Bob protects the confidentiality of the communication and the process's integrity.
4. **Basis disclosure**: Following the measurement step, Bob and Alice reveal the bases used for each qubit. This disclosure allows both parties to match their measurement bases which helps in the creation of a shared secret key. Incompatible bases and their bit values will be removed.
5. **Verification**: To confirm that the key was successfully transmitted, Bob and Alice share random samples of their keys. By comparing these samples, they can ensure that the transmission is accurate within a very small range of error. This verification stage is essential for determining the reliability of the quantum key distribution process.

Table 6 provides a concise summary of how knowledge is distributed throughout the QKD BB84 protocol.

Table 6. Knowledge exchange protocol.

Steps	Alice's Information	Over Eve's Communication Channel	Bob's Information
1	Alice's classical bits Alice's choice of bases		
2	Qubits	Qubits	Qubits
3			Bob's chosen bases Bob's measurement results
4	Bob's chosen bases	Alice's chosen bases Bob's chosen bases	Alice's chosen bases
5	Alice's encrypted key (noisy)		Bob's encrypted key (noisy)
6	Bob's measurement sample Alice's measurement sample	Bob's measurement sample Alice's measurement sample	Bob's measurement sample Alice's measurement sample
7	Shared secret key		Shared secret key

In the QKD BB84 protocol, the classical channel simulation plays a crucial role and it works after Bob's measurement, which is the third step. At this stage, Bob successfully receives the qubits and obtains the measurement outcomes. During this pivotal phase, Alice and Bob collaborate to establish both public and private keys, enhancing the security of their digital communications.

Moving forward, this research centers on the CRYSTALS-Kyber algorithm, the renowned KEM within the realm of PQC. The primary objective is to securely exchange a symmetric key between Alice and Bob, leveraging CRYSTALS-Kyber as a KEM. This key will subsequently facilitate AES 256-bit encryption between the two parties.

Expanding the scope of our research, we delve into digital signature mechanisms in the context of quantum computing. We assess two significant contenders in this domain, CRYSTALS-Dilithium and Falcon. The evaluation centers on their ability to ensure message integrity and authenticity within a quantum computing environment, as well as their

resistance to quantum attacks. In this stage, we want to check the performance of both PQC digital signature schemes to find the best combination of our authenticated encryption scheme for QKD.

3.3. Classical Channel Simulation

As can be seen in Figure 1, in a precisely designed series of actions, Alice and Bob independently generate their chosen digital signature public and private keys. They right away exchange the public keys between them, and Alice goes a step further by generating Kyber's public and private keys. She applies her chosen digital signature algorithm to sign her public key, marking a critical stage in the secure key exchange process. Bob then takes on the role of the verifier, receiving Alice's Kyber public key and employing the designated digital signature algorithm to authenticate its origin, thereby confirming Alice's identity and preserving the integrity of the transmitted data. Demonstrating cryptographic sophistication, Bob generates a robust 256-bit random string. He encapsulates this string using Alice's public key and appends a digital signature, ensuring the security of the entire package, which represents a securely determined secret dispatched back to Alice. Upon receipt of this encapsulated key packet, Alice initiates a series of cryptographic operations to validate its authenticity. She subjects the encapsulated data to rigorous verification processes, all relying on the public key associated with the chosen digital signature algorithm. Once the integrity of the data is confirmed, she proceeds to decrypt the encapsulated string with her private Kyber key. Through these intricate algorithms and complex mechanisms, including encapsulation, signing, and verification, Alice and Bob successfully share a reliable 256-bit key, serving as the cornerstone of their secure digital communication. This key ensures data confidentiality, authentication, and trustworthiness. Finally, with the shared key in hand, both parties can employ AES symmetric encryption with utmost confidence in the security and privacy of their information exchange, thanks to their meticulous orchestration of cryptographic protocols, which has forged an unbreakable bond of trust in their digital communications.

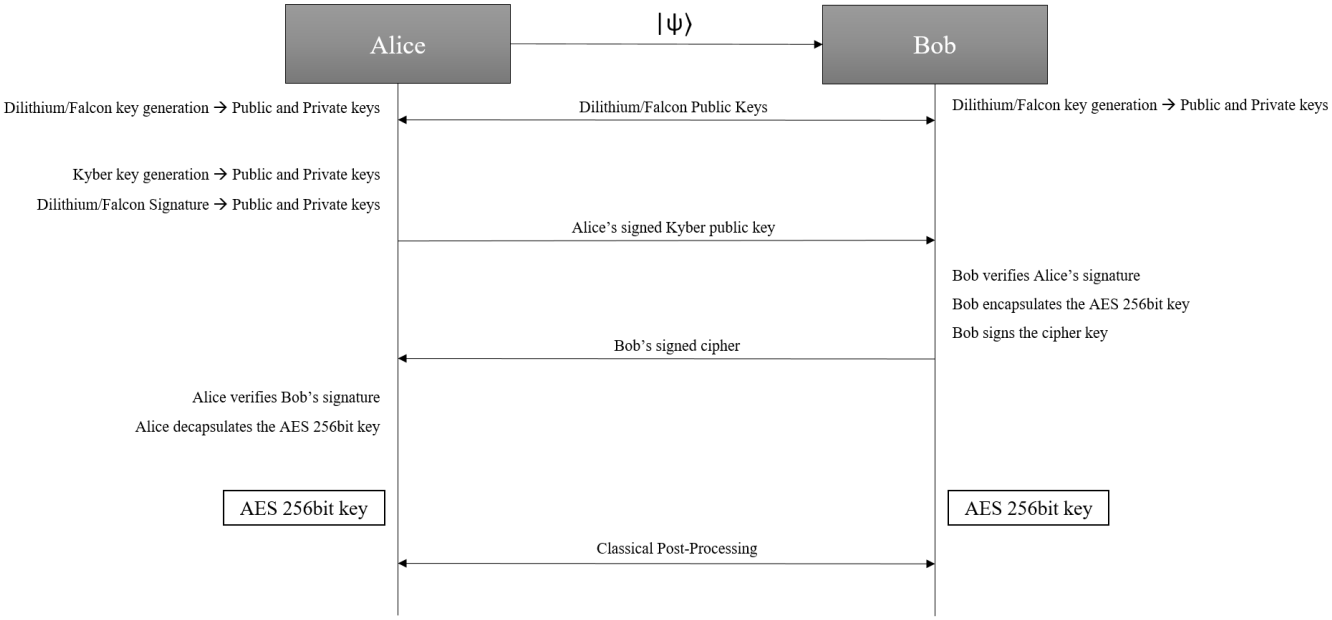

Figure 1. Post-Quantum cryptography process in the QKD classic channel.

3.4. Implementation Details

To implement PQC algorithms, the project applies several libraries, include the `GiacomoPope/kyber-py` [54] for Kyber, `GiacomoPope/dilithium-py` [55] for Dilithium, and `tprest/falcon.py` [56] for Falcon. The PQC algorithms are designed and executed with

the `Qiskit` library [57], which manipulates quantum circuits and provides the simulation for quantum protocol and algorithms. This includes operations such as quantum key generation, encoding, and decoding qubits. The `PyCrypto` library [58] for AES encryption. Execution timing is performed with `time` library [59] and performance evaluation is conducted using `NumPy` [60] for numerical computations. To test this setup, including key generation, encapsulation, decapsulation, signing, and verification, the project was executed on a Windows 10 64-bit system. The hardware configuration consists of an Intel(R) Core(TM) i7-8750H CPU @ 2.20 GHz and 12 GB RAM.

4. Results

Table 7 and Figures 2–4 show a wide range of performance benchmarks, providing an extensive analysis of the effectiveness and efficiency of three well-known cryptographic schemes: CRYSTALS-Kyber, CRYSTALS-Dilithium, and Falcon. These resources move beyond simple comparisons, delving into the details of major generation times, and exposing insights critical for informed decision making. The evident pattern of increasing key generation times with increasing security levels gives meaningful information about the challenging balancing of security robustness and computational needs. This discovery highlights the dynamic nature of cryptographic algorithms and how they respond to increased security needs. The shift from Dilithium to Falcon is particularly notable as large alterations in key generation times become apparent. This highlights the different structural complexities built into each signature method. Such detailed information helps administrators to navigate the complicated field of cryptographic options, and it also offers a mindful selection that meets not just severe security requirements but also realistic performance limitations.

Table 7. Key generation times (ms).

Security Level	Key Generation (ms)
Kyber 512	31.2
Kyber 768	78.1
Kyber 1024	93.72
Dilithium 2	62.42
Dilithium 3	93.73
Dilithium 5	171.87
Falcon 256	6606.39
Falcon 512	10,016.76
Falcon 1024	53,601.41

In basic terms, the benchmarks serve as both an in-depth guide and a strategy that shows the numerous choices related to cryptographic decisions. By providing a more in-depth understanding of these compromises, administrators obtain essential insights into the dynamic connection between security and performance concerns. This detailed perspective makes it easier to select cryptographic algorithms that are perfectly suited to the specific security and performance needs of a given application or system. Essentially, these benchmarks function as guidelines which lead the path to making optimal cryptographic decisions in the continuously changing field of digital security.

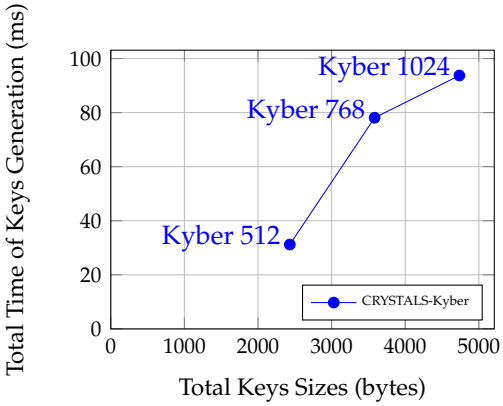

Figure 2. Comparison of CRYSTALS-Kyber key generation time and sizes.

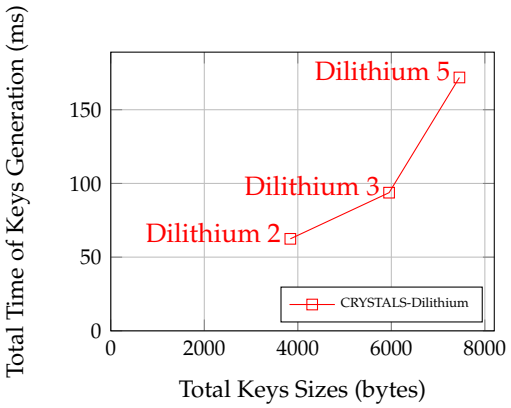

Figure 3. Comparison of CRYSTALS-Dilithium key generation time and sizes.

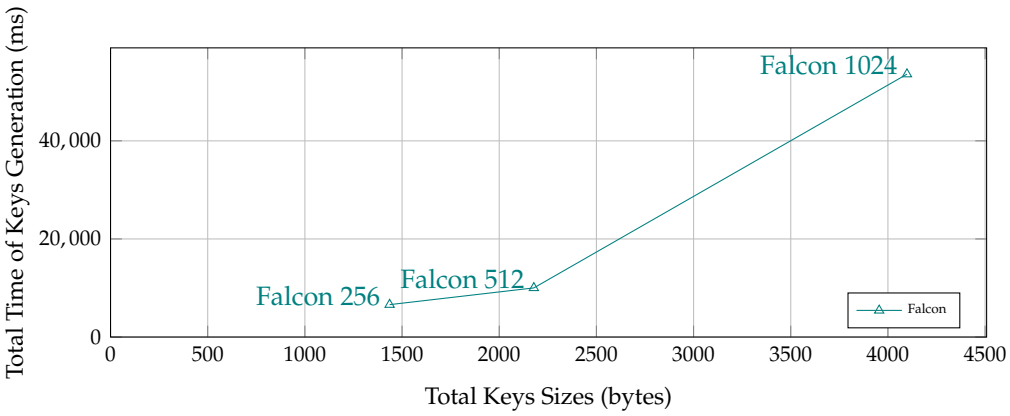

Figure 4. Comparison of Falcon key generation time and sizes.

In Figure 5, signature sizes, we have a comparison of the sizes of cryptographic signatures for different algorithms and configurations. The x-axis represents our various cryptographic algorithms and their key sizes, including Falcon 256, Falcon 512, Falcon 1024, Dilithium 2, Dilithium 3, and Dilithium 5. The y-axis represents the size of the signatures in bytes. Falcon 256 has the smallest signature size at 356 bytes, followed by Falcon 512 at 666 bytes, and Falcon 1024 with the largest signature size at 1280 bytes. On the other hand, the Dilithium signature sizes, which are Dilithium 2, Dilithium 3, and Dilithium 5, are larger, beginning with 2420 to 4595 bytes. This figure provides the efficiency of different signature schemes in terms of signature size. As the security level increases, the size increases as well. Figure 6, ciphertext sizes for CRYSTALS-Kyber configurations, represents

the sizes of ciphertexts generated by all three security levels of the CRYSTALS-Kyber cryptographic algorithm including Kyber 512, Kyber 768, and Kyber 1024. Examining these data, we can see that Kyber 512 generates ciphertexts of size 768 bytes, Kyber 768 produces ciphertexts of size 1088 bytes, and Kyber 1024 results in ciphertexts of size 1568 bytes. This figure illustrates how the choice of Kyber security level impacts the size of ciphertexts, which is essential for assessing the trade-off between security and efficiency in cryptographic applications.

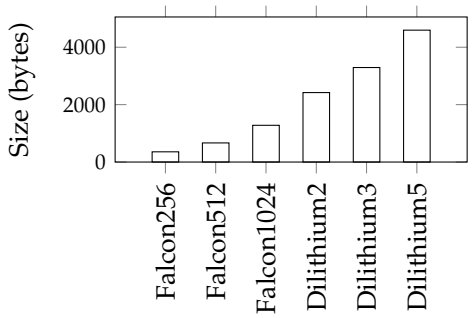

Figure 5. Signature sizes.

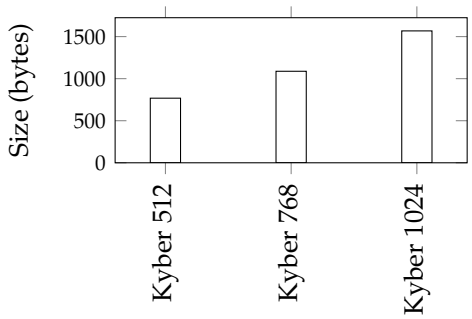

Figure 6. Ciphertext sizes.

Significant differences in key size, cipher size, and signature size among algorithms arise from their principles and design. In CRYSTALS-Kyber, the public key consists of a matrix and some random values used in the encryption process. This public key size is quite large. The ciphertext is generated during the encryption process and includes some compressed values derived from the public key and the message. Compression techniques are used to reduce the size of these values, which makes the ciphertext smaller than the public key. CRYSTALS-Dilithium also uses lattice-based cryptography. It achieves a lower signature size than the key size by using structured lattices. Falcon is based on the NTRU lattice problem and uses the FFT for optimization. This allows Falcon to achieve very compact signatures and smaller key sizes compared to other lattice-based algorithms.

Figures 7–9 provide a detailed examination of the encapsulation time of CRYSTALS-Kyber across various signing algorithms. These figures illustrate the time, measured in milliseconds, needed to encapsulate data for different security levels. It is important to observe that as security levels increase, the time required for this operation also increases. This increase in time reflects the enhanced security and encryption offered by these algorithms. Achieving the right balance between security and time efficiency is crucial for specific use cases.

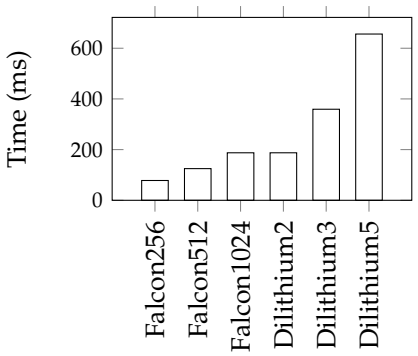

Figure 7. Kyber 512 encapsulation.

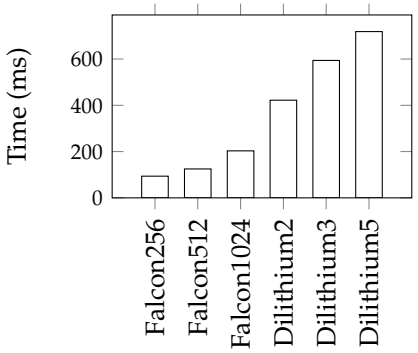

Figure 8. Kyber 768 encapsulation.

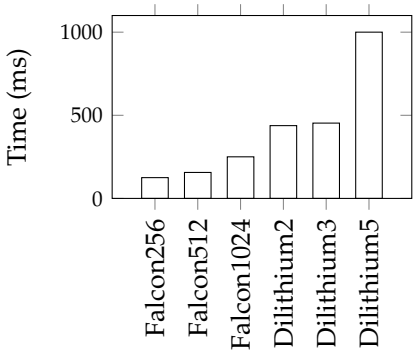

Figure 9. Kyber 1024 encapsulation.

Figures 10–12 offer valuable insights into the timeline for decapsulation and verification, focusing on the complex process of decrypting and verifying data secured using CRYSTALS-Kyber and different signing algorithms. Notably, a moderate increase in time becomes evident as security levels rise, offering a trade-off for improved protection. It is worth emphasizing that when utilizing CRYSTALS-Dilithium, the time investment remains considerably lower compared to both encapsulation and signing. This illustrates its efficiency, a crucial consideration tailored to a variety of use cases and security requirements.

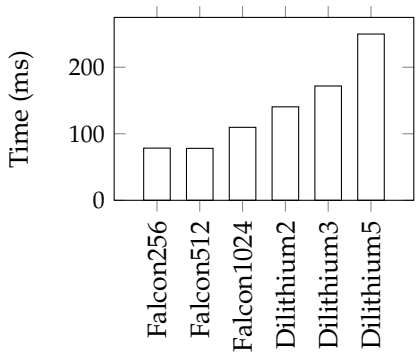

Figure 10. Kyber 512 decapsulation.

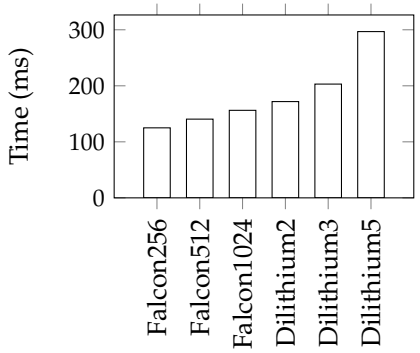

Figure 11. Kyber 768 decapsulation.

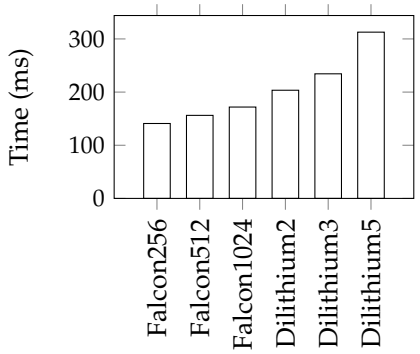

Figure 12. Kyber 1024 decapsulation.

Figures 13 and 14 illustrate the performance of three Kyber post-quantum cryptographic variants (Kyber 512, Kyber 768, and Kyber 1024) in terms of total data size (x-axis in bytes) and processing time (y-axis in milliseconds). Each point on the graph represents a digital signature that is combined with the specified Kyber variant. In Figure 13, starting from Falcon 256, the graph progresses to Falcon 512 and Falcon 1024, demonstrating the variations in processing time concerning different signature sizes. In Figure 14, the focus shifts to Dilithium, with points corresponding to Dilithium 2, Dilithium 3, and finally Dilithium 5. This comprehensive visualization provides insights into the trade-offs between signature size and processing time for each cryptographic scheme. For Kyber 512, processing time increases with data size, and a similar trend is observed for Kyber 768 and Kyber 1024. These figures are valuable documents for selecting the appropriate Kyber variant tailored to specific application needs, allowing for a well-informed decision that balances security and efficiency considerations. Kyber 512 offers quicker processing for smaller data sizes, while Kyber 1024 provides higher security at slightly longer processing times.

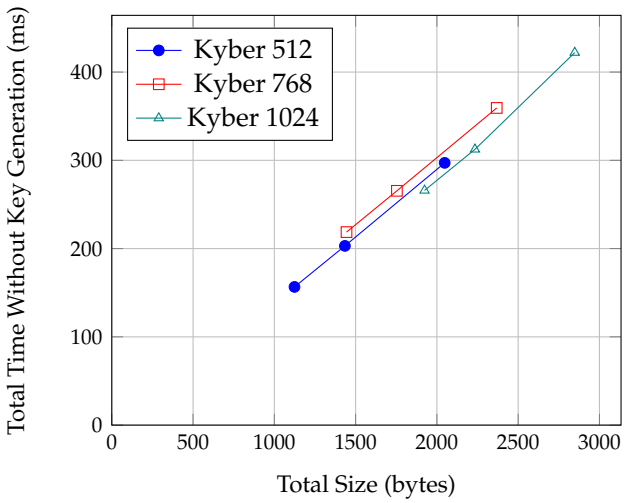

Figure 13. Performance comparison of authenticated-encryption schemes (using Falcon).

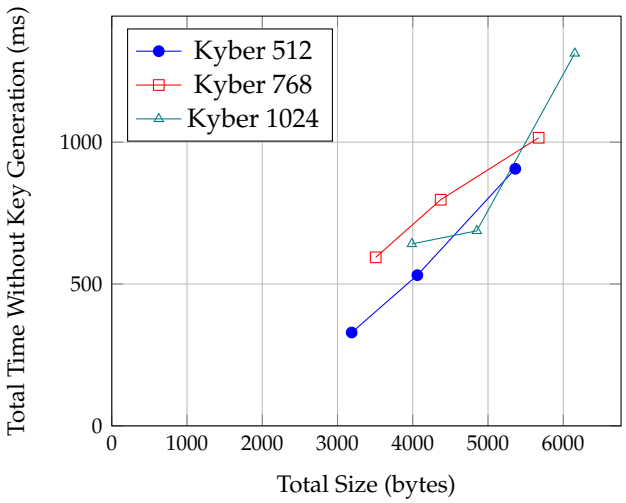

Figure 14. Performance comparison of authenticated-encryption schemes (using CRYSTALS-Dilithium).

Additionally, Tables 8–11 offer an in-depth overview of all cryptographic settings, showing the complex relationships between cipher sizes, signature sizes, and the time needed for key operations. As we peruse the data, we discern intriguing patterns and trade-offs. For instance, increasing the security level of cryptographic algorithms, such as transitioning from Kyber 512 to Kyber 1024, leads to larger cipher and signature sizes. However, this enhancement in security comes at a cost, as both encapsulation and signing times also increase. Notably, the signature size experiences substantial variations as we switch between Dilithium and Falcon, showcasing how different signature algorithms can drastically alter the size of the digital seal appended to messages. In contrast, the cipher size remains fairly stable within each security level of the Kyber algorithm, emphasizing that the core encryption method has a more consistent impact on the size of the encrypted data. This insight into the relationship between signature schemes and signature size versus the influence of encryption algorithms on cipher size provides valuable guidance for tailoring cryptographic solutions to specific security and efficiency requirements.

To have a logical comparison between PQC schemes and RSA, we choose to compare the three security levels of PQC algorithms with RSA key sizes of 3072, 7680, and 15,360 bits. In order to achieve the optimal balance between the security and performance of PQC schemes, we used RSA as secure KEM and digital signature to gain similar results to previous ones. Table 12 shows the performance of different key sizes, which also demonstrates the security of RSA.

Table 8. Encapsulation and signing, decapsulation, and verification times.

Cipher Combination	Encapsulation and Signing (ms)	Decapsulation and Verification (ms)
Kyber 512 and Falcon 256	78.13	78.44
Kyber 768 and Falcon 256	93.72	124.97
Kyber 1024 and Falcon 256	124.97	140.92
Kyber 512 and Falcon 512	124.97	78.07
Kyber 768 and Falcon 512	124.94	140.52
Kyber 1024 and Falcon 512	156.21	156.21
Kyber 512 and Falcon 1024	187.45	109.68
Kyber 768 and Falcon 1024	203.11	156.23
Kyber 1024 and Falcon 1024	250.19	171.86
Kyber 512 and Dilithium 2	187.45	140.52
Kyber 768 and Dilithium 2	422.10	171.84
Kyber 1024 and Dilithium 2	437.77	203.48
Kyber 512 and Dilithium 3	359.29	171.83
Kyber 768 and Dilithium 3	593.94	203.07
Kyber 1024 and Dilithium 3	453.01	234.37
Kyber 512 and Dilithium 5	656.02	249.94
Kyber 768 and Dilithium 5	718.61	296.77
Kyber 1024 and Dilithium 5	1000.07	312.75

Table 9. Cipher sizes.

Cipher	Cipher Size (bytes)
Kyber 512	768
Kyber 768	1088
Kyber 1024	1568

Table 10. Signature sizes.

Cipher	Signature Size (bytes)
Falcon 256	356
Falcon 512	666
Falcon 1024	1280

Table 11. Signature sizes.

Cipher	Signature Size (bytes)
Dilithium 2	2420
Dilithium 3	3293
Dilithium 5	4595

Table 12. RSA performance.

Key Size	Key Generation (ms)	Encapsulation and Signing (ms)	Decapsulation and Verification (ms)
3072	334.34	16.76	15.66
7680	4461.05	49.21	33.69
15,360	39,045.80	199.52	189.16

From these results of RSA performance and security levels, we choose Kyber 512 with Dilithium 3 as the optimal combination that has a great balance between security and speed. When comparing the choice of Dilithium 3 with Kyber 512 against RSA and Falcon combinations, several factors influence the decision. RSA offers dual functionality as both a digital signature and a KEM, with established security but increasingly longer key

generation times at higher security levels (up to 39,045.80 ms for RSA-15360). Falcon, while efficient in operational times, requires extensive key generation times (up to 10,016.76 ms for Falcon 512), which makes it less suitable for applications requiring quick key setup. In contrast, Kyber 512 with Dilithium 3 maintains a balance by providing a total key generation time of 124.93 ms (31.2 ms for Kyber 512 and 93.73 ms for Dilithium 3). This combination also offers reasonable operational times with encapsulation and signing at 359.29 ms and decapsulation and verification at 171.83 ms. As a result, Kyber 512 and Dilithium 3 present a compelling option that provides the best combination of efficient key generation, reasonable operational times, and high security requirements, making them an acceptable substitute to both RSA and Falcon in post-quantum cryptographic applications.

5. Discussion

As we discussed, in the future, when quantum computers could represent an issue, post-quantum encryption will depend on robust and computationally secure methods to protect sensitive data. The standardization of algorithms by NIST has led to the attention and acceptance of CRYSTALS-Kyber, CRYSTALS-Dilithium, and Falcon. However, NIST advises CRYSTALS-Dilithium as the preferred option for digital signatures and CRYSTALS-Kyber as the first PQC key exchange technique. Lattice-based cryptography issues are at the core of CRYSTALS-Kyber, CRYSTALS-Dilithium, and Falcon.

In our results in comparing post-quantum cryptographic solutions, the combination of Kyber 512 and Dilithium 3 proved superior when balancing key generation time, operational performance, and security. The performance of RSA declined significantly with larger key sizes; however, Falcon was efficient in operating times but has long key creation times (up to 10,016.76 ms for Falcon 512), making it unsuitable for rapid key setup. Kyber 512 and Dilithium 3 provide an optimal option, with a total key generation time of 124.93 ms, encapsulation and signature at 359.29 ms, and decapsulation and verification at 171.83 ms.

Despite their robustness, there remains a concern that these algorithms may eventually expose vulnerabilities, emphasizing the need for continuous scrutiny in the field. QKD, on the other hand, is secure based on the principles of quantum mechanics. This raises the question of whether using technically safe authentication methods could negatively impact the security of QKD. It is crucial to clarify that the authentication strategy used in QKD only needs to be temporarily secure. Therefore, there is often no need for concern regarding the computational security of the encapsulation and authentication procedures.

If the encapsulation or authentication procedures were compromised during the use of the QKD protocol, a man-in-the-middle attack could be executed successfully. However, even if such a breach occurred after the key exchange, the security of the quantum-distributed symmetric encryption keys would remain intact. In the rare instance that the encapsulation and authentication methods employed in QKD become vulnerable to an attacker with significant computational resources, a straightforward countermeasure would be to update the algorithm to a more secure one. This could be achieved without ever placing the keys produced by QKD at risk.

This shows the importance of flexibility and adaptability in cryptographic protocols. As computational power increases and new vulnerabilities are discovered, the ability to quickly update and improve encryption and authentication methods becomes crucial. The concept of forward secrecy (FS), also known as perfect forward secrecy (PFS), where past communications remain secure even if current keys are compromised, is particularly relevant here. By ensuring that each key exchange session is independently secure, we can protect the integrity of encrypted data over time.

Moreover, the use of computationally secure encapsulation and authentication in QKD does not compromise its security, rendering privacy amplification unnecessary. This illustrates an appropriate justification for integrating post-quantum cryptographic methods with QKD, enhancing the overall security framework without introducing new vulnerabilities.

6. Conclusions and Future Work

In this research, we introduced a new approach for using PQC for the classic channel of QKD and presented an optimal combination for both encryption and authentication. The choice of the best security level for cryptographic algorithms like CRYSTALS-Kyber, CRYSTALS-Dilithium, and Falcon ultimately depends on striking the right balance between the required level of security and the performance and limits on resources set by the intended use case. These cryptographic techniques each provide a range of security levels that are differentiated by many factors, such as key sizes, signature sizes, and computing demands. Higher security levels, including Kyber 1024, Dilithium 5, and Falcon 1024, often offer more powerful security assurances, making them desirable options for applications where strong security is crucial. It is important to understand, nevertheless, that this increased security frequently results in larger signatures and slower cryptographic procedures. The "best" security level must, thus, be specifically customized for the application in question's unique security needs and performance limits. For instance, Kyber 512 and Dilithium 3 represent well-balanced choice, offering a commendable compromise between security and performance. They can be widely adopted and considered secure for most practical purposes. However, we should consider that the Falcon worked well in terms of signing and verification but its performance was not good in the key generation phase.

There are various unexplored options for future investigation. Future research could delve deeper into symmetric encryption methods in order to gain a more in-depth understanding of AES's security and its future among the other schemes. Additionally, an investigation into quantum secure communication in quantum repeaters is also significant. This study creates the groundwork for future study and allows researchers interested in the implementation of PQC schemes to contribute to the continuous advancement of knowledge in this field.

Author Contributions: Conceptualization, F.R.G., M.Y. and N.E.; methodology, F.R.G. and M.Y.; software, F.R.G. and M.Y; validation, F.R.G., N.E. and M.Y.; formal analysis, F.R.G. and M.Y.; investigation, F.R.G., N.E. and M.Y.; resources, M.Y. and N.E.; data curation, F.R.G.; writing—original draft preparation, F.R.G.; writing—review and editing, F.R.G., N.E., Y.A. and M.Y.; visualization, F.R.G., N.E. and M.Y.; supervision, M.Y.; project administration, N.E. and M.Y.; funding acquisition, N.E. All authors have read and agreed to the published version of the manuscript.

Funding: Glasgow Caledonian University provided the funding for open access for this research.

Data Availability Statement: The code repository for this project can be accessed by clicking https://github.com/FarshadRahimiGhashghaei/Research-Project (accessed on 27 February 2024). Within the `main.py` file, three PQC algorithms are integrated: Kyber for key encapsulation, and Dilithium and Falcon for authentication, all within the QKD BB84 simulation. This integration enhances the code structure and lets us evaluate different algorithms within our cryptosystem. Each component (`Kyber.py`, `Dilithium.py`, and `Falcon.py`) is used within the `main.py` file.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Iqbal, S.S.; Zafar, A. Enhanced Shor's algorithm with quantum circuit optimization. *Int. J. Inf. Technol.* **2024**, *16*, 2725–2731.
2. Biswas, S.; Das, P. Analysis of Quantum Cryptology and the RSA Algorithms Defense against Attacks Using Shor's Algorithm in a Post Quantum Environment. In Proceedings of the International Conference on Computational Intelligence in Communications and Business Analytics, Kalyani, India, 27–28 January 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 72–87.
3. Larasati, H.T.; Kim, H. Quantum cryptanalysis landscape of shor's algorithm for elliptic curve discrete logarithm problem. In Proceedings of the Information Security Applications: 22nd International Conference, WISA 2021, Jeju Island, Repulic of Korea, 11–13 August 2021; Revised Selected Papers 22; Springer: Berlin/Heidelberg, Germany, 2021; pp. 91–104.
4. Malina, L.; Ricci, S.; Dzurenda, P.; Smekal, D.; Hajny, J.; Gerlich, T. Towards practical deployment of post-quantum cryptography on constrained platforms and hardware-accelerated platforms. In *Innovative Security Solutions for Information Technology and Communications*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 109–124.
5. Mitra, S.; Jana, B.; Bhattacharya, S.; Pal, P.; Poray, J. Quantum cryptography: Overview, security issues and future challenges. In Proceedings of the 2017 4th International Conference on Opto-Electronics and Applied Optics (Optronix), Kolkata, India, 2–3 November 2017.

6. Asif, R. Post-quantum cryptosystems for Internet-of-Things: A survey on lattice-based algorithms. *IoT* **2021**, *2*, 71–91.
7. Liu, F.; Zheng, Z.; Gong, Z.; Tian, K.; Zhang, Y.; Hu, Z.; Li, J.; Xu, Q. A survey on lattice-based digital signature. *Cybersecurity* **2024**, *7*, 7.
8. Balamurugan, C.; Singh, K.; Ganesan, G.; Rajarajan, M. Post-quantum and code-based cryptography—Some prospective research directions. *Cryptography* **2021**, *5*, 38.
9. Deneuville, J.C. *Code-Based Cryptography: 10th International Workshop, CBCrypto 2022, Trondheim, Norway, May 29–30, 2022, Revised Selected Papers*; Springer Nature: Berlin/Heidelberg, Germany, 2023; Volume 13839.
10. Li, L.; Lu, X.; Wang, K. Hash-based signature revisited. *Cybersecurity* **2022**, *5*, 13.
11. Mironov, I. Hash functions: Theory, attacks, and applications. *Microsoft Res. Silicon Val. Campus* **2005**, 1–22.
12. Calderini, M.; Caminata, A.; Villa, I. A new multivariate primitive from CCZ equivalence. *arXiv* **2024**, arXiv:2405.20968.
13. Billet, O.; Ding, J. Overview of cryptanalysis techniques in multivariate public key cryptography. In *Gröbner Bases, Coding, and Cryptography*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 263–283.
14. Yalamuri, G.; Honnavalli, P.; Eswaran, S. A review of the present cryptographic arsenal to deal with post-quantum threats. *Procedia Comput. Sci.* **2022**, *215*, 834–845.
15. Nejatollahi, H.; Dutt, N.; Ray, S.; Regazzoni, F.; Banerjee, I.; Cammarota, R. Post-quantum lattice-based cryptography implementations. *ACM Comput. Surv.* **2019**, *51*, 1–41.
16. Heisenberg, W. *The Actual Content of Quantum Theoretical Kinematics and Mechanics*; National Academy of Sciences: Washington, DC, USA, 1983; NAS 1.15: 77379.
17. Wootters, W.; Zurek, W. A single quantum cannot be cloned. *Nature* **1982**, *299*, 802–803.
18. Diamanti, E.; Lo, H.K.; Qi, B.; Yuan, Z. Practical challenges in quantum key distribution. *NPJ Quantum Inf.* **2016**, *2*, 16025.
19. Li, K.; Cai, Q. Practical security of RSA against NTC-architecture quantum computing attacks. *Int. J. Theor. Phys.* **2021**, *60*, 2733–2744.
20. Bennett, C.H.; Brassard, G. Quantum cryptography: Public key distribution and coin tossing. *Theor. Comput. Sci.* **2014**, *560*, 7–11.
21. Bisheh-Niasar, M.; Azarderakhsh, R.; Mozaffari-Kermani, M. Instruction-set accelerated implementation of crystals-kyber. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2021**, *68*, 4648–4659.
22. Yao, K.; Kundi, D.E.S.; Wang, C.; O'Neill, M.; Liu, W. Towards crystals-kyber: A M-LWE cryptoprocessor with area-time trade-off. In Proceedings of the 2021 IEEE International Symposium on Circuits and Systems (ISCAS), Daegu, Repulic of Korea, 22–28 May 2021.
23. Ducas, L.; Kiltz, E.; Lepoint, T.; Lyubashevsky, V.; Schwabe, P.; Seiler, G.; Stehlé, D. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2018**, *1*, 238–268.
24. Moody, D. Fast Fourier Sampling over NTRU Lattices Digital Signature Standard. Availble online: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.206.pdf (accessed on 23 January 2024).
25. Soni, D.; Basu, K.; Nabeel, M.; Aaraj, N.; Manzano, M.; Karri, R. *Hardware Architectures for Post-Quantum Digital Signature Schemes*; Springer: Berlin/Heidelberg, Germany, 2021.
26. Inamori, H.; Lütkenhaus, N.; Mayers, D. Unconditional security of practical quantum key distribution. *Eur. Phys. J. D* **2007**, *41*, 599–627.
27. Gleim, A.V.; Egorov, V.I.; Nazarov, Y.V.; Smirnov, S.V.; Chistyakov, V.V.; Bannik, O.I.; Anisimov, A.A.; Kynev, S.M.; Ivanova, A.E.; Collins, R.J.; et al. Secure polarization-independent subcarrier quantum key distribution in optical fiber channel using BB84 protocol with a strong reference. *Opt. Express* **2016**, *24*, 2619.
28. Ghamdi-Al, A.B.; Sulami-Al, A.; Aljahdali, A.O. On the security and confidentiality of quantum key distribution. *Secur. Priv.* **2020**, *3*, e111.
29. Padamvathi, V.; Vardhan, B.V.; Krishna, A.V.N. Quantum cryptography and quantum key distribution protocols: A survey. In Proceedings of the 2016 IEEE 6th International Conference on Advanced Computing (IACC), Bhimavaram, India, 27–28 February 2016.
30. Jha, M.S.; Maity, S.K.; Nirmal, M.K.; Krishna, J. A survey on quantum cryptography and quantum key distribution protocols. *Int. J. Adv. Res. Ideas Innov. Technol.* **2019**, *5*, 144–147.
31. Patel, N.A.; Patel, H.B. Analysis of network performance using aspect of quantum cryptography. *Int. J. Comput. Inf. Eng.* **2019**, *13*, 496–499.
32. Reddy, M.S.; Mohan, B.C. Comprehensive Analysis of BB84, A Quantum Key Distribution Protocol. *arXiv* **2023**, arXiv:2312.05609.
33. Huang, J.; Wang, Y.; Wang, H.; Li, Z.; Huang, J. Man-in-the-middle attack on BB84 protocol and its defence. In Proceedings of the 2009 2nd IEEE International Conference on Computer Science and Information Technology, Beijing, China, 8–11 August 2009.
34. Yesina, M.V.; Ostrianska, Y.V.; Gorbenko, I.D. Status report on the third round of the NIST post-quantum cryptography standardization process. *Radiotekhnika* **2022**, 75–86. [CrossRef]
35. Bos, J.; Ducas, L.; Kiltz, E.; Lepoint, T.; Lyubashevsky, V.; Schanck, J.M.; Schwabe, P.; Seiler, G.; Stehlé, D. Crystals—Kyber: A CCA-secure module-lattice-based KEM. In Proceedings of the 2018 IEEE European Symposium on Security and Privacy (EuroS&P), London, UK, 24–26 April 2018.
36. Jati, A.; Gupta, N.; Chattopadhyay, A.; Sanadhya, S.K. A configurable crystals-kyber hardware implementation with side-channel protection. *ACM Trans. Embed. Comput. Syst.* **2024**, *23*, 1–25.

37. Ni, Z.; Khalid, A.; O'Neill, M.; Liu, W. HPKA: A High-Performance CRYSTALS-Kyber Accelerator Exploring Efficient Pipelining. *IEEE Trans. Comput.* **2023**, *72*, 3340–3353.
38. Seyhan, K.; Akleylek, S. Indistinguishability under adaptive chosen-ciphertext attack secure double-NTRU-based key encapsulation mechanism. *PeerJ Comput. Sci.* **2023**, *9*, e1391. [CrossRef]
39. Sanal, P.; Karagoz, E.; Seo, H.; Azarderakhsh, R.; Mozaffari-Kermani, M. Kyber on ARM64: Compact implementations of Kyber on 64-bit ARM cortex-A processors. In *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*; Springer International Publishing: Cham, Switzerland, 2021; pp. 424–440.
40. Zhou, Z.; He, D.; Liu, Z.; Luo, M.; Choo, K.K.R. A software/hardware co-design of crystals-dilithium signature scheme. *ACM Trans. Reconfigurable Technol. Syst.* **2021**, *14*, 1–21.
41. Qiao, Z.; Liu, Y.; Zhou, Y.; Ming, J.; Jin, C.; Li, H. Practical public template attacks on crystals-dilithium with randomness leakages. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 1–14.
42. Beckwith, L.; Nguyen, D.T.; Gaj, K. High-performance hardware implementation of crystals-dilithium. In Proceedings of the 2021 International Conference on Field-Programmable Technology (ICFPT), Auckland, New Zealand, 6–10 December 2021; pp. 1–10.
43. Soni, D.; Ducas, L.; Kiltz, E.; Lepoint, T.; Schwabe, P.; Seiler, G.; Stehlé, D.; Bai, S. Crystals-dilithium. In *Hardware Architectures for Post-Quantum Digital Signature Schemes*; Springer: Cham, Switzerland, 2020; pp. 13–30. [CrossRef]
44. Nguyen, D.T.; Gaj, K. Fast falcon signature generation and verification using armv8 neon instructions. In Proceedings of the Progress in Cryptology—AFRICACRYPT 2023, Sousse, Tunisia, 19–21 July 2023; pp. 417–441. [CrossRef]
45. Seo, E.Y.; Kim, Y.S.; Lee, J.W.; No, J.S. Peregrine: Toward Fastest FALCON Based on GPV Framework. Cryptology ePrint Archive, Paper 2022/1495, 2022. Available online: https://eprint.iacr.org/2022/1495 (accessed on 4 June 2024).
46. Fouque, P.A.; Hoffstein, J.; Kirchner, P.; Lyubashevsky, V.; Pornin, T.; Prest, T.; Ricosset, T.; Seiler, G.; Whyte, W.; Zhang, Z. Falcon: Fast-Fourier lattice-based compact signatures over NTRU. *Submiss. NIST's-Post-Quantum Cryptogr. Stand. Process.* **2018**, *36*, 1–75.
47. Chen, Y. Quantum Algorithms for Lattice Problems. Cryptology ePrint Archive, Paper 2024/555, 2024. Available online: https://eprint.iacr.org/2024/555 (accessed on 4 June 2024).
48. Ahn, J.; Kwon, H.Y.; Ahn, B.; Park, K.; Kim, T.; Lee, M.K.; Kim, J.; Chung, J. Toward quantum secured distributed energy resources: Adoption of post-quantum cryptography (pqc) and quantum key distribution (qkd). *Energies* **2022**, *15*, 714.
49. Wang, L.J.; Zhang, K.Y.; Wang, J.Y.; Cheng, J.; Yang, Y.H.; Tang, S.B.; Yan, D.; Tang, Y.L.; Liu, Z.; Yu, Y.; et al. Experimental authentication of quantum key distribution with post-quantum cryptography. *NPJ Quantum Inf.* **2021**, *7*, 67.
50. Yang, Y.H.; Li, P.Y.; Ma, S.Z.; Qian, X.C.; Zhang, K.Y.; Wang, L.J.; Zhang, W.L.; Zhou, F.; Tang, S.B.; Wang, J.Y.; et al. All optical metropolitan quantum key distribution network with post-quantum cryptography authentication. *Opt. Express* **2021**, *29*, 25859–25867.
51. Cross, A. The IBM Q experience and QISKit open-source quantum computing software. In Proceedings of the APS March Meeting Abstracts, Los Angeles, CA, USA, 5–9 March 2018; Volume 2018, p. L58-003.
52. Pereira, M.; Currás-Lorenzo, G.; Navarrete, Á.; Mizutani, A.; Kato, G.; Curty, M.; Tamaki, K. Modified BB84 quantum key distribution protocol robust to source imperfections. *Phys. Rev. Res.* **2023**, *5*, 023065.
53. Boyer, M.; Liss, R.; Mor, T. Composable security of generalized BB84 protocols against general attacks. *arXiv* **2022**, arXiv:2208.12154.
54. Pope, G. GiacomoPope/kyber-py. 2024. Available online: https://github.com/GiacomoPope/kyber-py (accessed on 4 June 2024).
55. Pope, G. GiacomoPope/dilithium-py. 2024. Available online: https://github.com/GiacomoPope/dilithium-py (accessed on 4 June 2024).
56. Prest, T. tprest/falcon.py. 2024. Available online: https://github.com/tprest/falcon.py (accessed on 4 June 2024).
57. ibm.com. Qiskit | IBM Quantum Computing. Available online: https://ibm.com/quantum/qiskit (accessed on 4 June 2024).
58. Litzenberger, D. pycrypto: Cryptographic Modules for Python. 2013. Available online: https://pypi.org/project/pycrypto/ (accessed on 4 June 2024).
59. Python Software Foundation. Time–Time Access and Conversions–Python 3.7.2 Documentation. 2000. Available online: https://docs.python.org/3/library/time.html (accessed on 4 June 2024).
60. Numpy. NumPy. 2009. Available online: https://numpy.org/ (accessed on 4 June 2024).