Scholarly Research Journal for Interdisciplinary Studies
Online ISSN 2278-8808, SJIF 2026 = 8.727
https://www.srjis.com/issues_data/297
Peer Reviewed & Refereed Journal, JULY-AUG 2026, Vol- 15/96

A COMBINATION OF BB84 QUANTUM KEY DISTRIBUTION AND AN IMPROVED SCHEME OF KYBER POST-QUANTUM CRYPTOSYSTEM

J Mani Ram
Department of Electronics and Communication Engineering PES University, Bengaluru, India
Santhameena S
Department of Electronics and Communication Engineering PES University, Bengaluru, India

Paper Received On: 1 AUG 2026
Peer Reviewed On: 16 AUG 2026
Published On: 01 SEPT 2026

Abstract

The emergence of quantum computing has exposed critical vulnerabilities in classical cryptographic systems such as RSA and ECC, which rely on problems like integer factorization and discrete logarithms. To address these challenges, this work proposes a hybrid cryptographic framework that integrates Quantum Key Distribution (QKD) with post-quantum encryption. Specifically, the BB84 protocol is employed for secure key generation, while the Kyber lattice-based cryptosystem provides efficient and quantum-resistant encryption. The system is evaluated under multiple attack scenarios including intercept-resend, noise injection, man-in-the-middle, photon loss, and replay attacks. Performance metrics such as Quantum Bit Error Rate (QBER), execution time, and key consistency are analyzed to assess robustness. Experimental results demonstrate that BB84 consistently achieves lower error rates compared to E91, and Kyber outperforms other post-quantum algorithms such as NTRU and SPHINCS+ in computational efficiency. The combined BB84–Kyber model delivers strong resilience against both classical and quantum adversaries, offering a practical balance between security and performance. This study highlights the importance of hybrid cryptographic approaches and provides a comparative framework that can guide the design of next-generation secure communication systems.

Index Terms: *Quantum Key Distribution, BB84, Kyber, Post-Quantum Cryptography, Hybrid Cryptosystem, Secure Communication*

I. INTRODUCTION

A. Limitations of Classical Cryptography

Classical cryptographic systems such as RSA and Elliptic Curve Cryptography (ECC) have been the backbone of secure communication for decades. Their security relies on the computational difficulty of problems like integer factorization and discrete logarithms. However, with the emergence of quantum algorithms such as Shor's, these problems can be solved in polynomial time, making RSA and ECC vulnerable. Beyond quantum threats, classical systems

Copyright © 2026, Scholarly Research Journal for Interdisciplinary Studies

also face side-channel attacks, brute-force attempts, and implementation flaws, which reduce their long-term reliability. As data exchange grows across finance, healthcare, and government sectors, the limitations of classical cryptography highlight the urgent need for more resilient solutions.

B. *Quantum Cryptography*

Quantum cryptography introduces a paradigm shift by using the principles of quantum mechanics to secure communication. Unlike classical methods that depend on mathematical hardness, quantum systems guarantee security through physical laws such as the Heisenberg Uncertainty Principle and the No-Cloning Theorem. Any attempt to intercept quantum states introduces detectable disturbances, enabling parties to identify eavesdropping attempts. Quantum Key Distribution (QKD) is the most prominent application, allowing two parties to establish a shared secret key with provable security.

1) BB84 Protocol: The BB84 protocol, proposed by Bennett and Brassard in 1984, is the first and most widely implemented QKD scheme. It uses photon polarization states across two bases — rectilinear and diagonal — to encode bits. Alice transmits photons with random polarization, while Bob measures them using randomly chosen bases. After transmission, they publicly compare bases and retain only matching results, forming a sifted key. Any eavesdropping attempt increases the Quantum Bit Error Rate (QBER), allowing detection. BB84 is valued for its simplicity, robustness, and suitability for practical deployment.

2) E91 Protocol: The E91 protocol, introduced by Ekert in 1991, is based on quantum entanglement and Bell's inequalities. Entangled photon pairs are distributed between Alice and Bob, and correlations in their measurements are used to generate keys. The violation of Bell's inequalities ensures that any external interference can be detected. While E91 provides strong theoretical guarantees and deeper ties to quantum mechanics, it is more complex to implement and highly sensitive to environmental noise. This makes it less practical compared to BB84, though it remains important for advancing entanglement-based cryptography.

C. *Post-Quantum Cryptography*

Post-quantum cryptography (PQC) focuses on algorithms that remain secure against quantum adversaries but operate on classical hardware. These schemes are designed around problems believed to be hard even for quantum computers, such as lattice problems, hash functions, and code-based constructions. PQC is critical because it can be deployed using existing infrastructure without requiring quantum communication channels.

1) Kyber: Kyber is a lattice-based key encapsulation mechanism built on the Learning With Errors (LWE) problem. It has been selected by NIST for standardization due to its strong balance of efficiency, security, and deterministic decapsulation behavior. Kyber offers small key sizes, fast computation, and strong resistance to quantum attacks, making it highly suitable for real-world applications.

2) NTRU: NTRU is another lattice-based cryptosystem that has been studied extensively for decades. It provides efficient encryption and decryption operations, but its larger key sizes and certain performance trade-offs make it less attractive compared to Kyber. Nevertheless, NTRU

remains a strong candidate in the PQC landscape due to its proven security and long history of analysis.

3) *SPHINCS+:* SPHINCS+ is a hash-based signature scheme designed to provide quantum-resistant authentication. It relies on the security of hash functions and does not depend on number-theoretic assumptions. While SPHINCS+ offers strong security guarantees, its computational overhead and slower performance make it more suitable for digital signatures than for encryption tasks.

D. *Need for Hybrid Cryptographic Systems*

Quantum cryptography ensures secure key distribution but lacks efficient encryption mechanisms, while PQC provides robust encryption but cannot detect eavesdropping during key exchange. A hybrid approach that integrates QKD with PQC combines the strengths of both paradigms. In this work, BB84 is used for secure key generation, and Kyber is employed for efficient encryption. This combination provides resilience against classical and quantum adversaries, achieving both low error rates and strong computational efficiency. The hybrid model is evaluated under multiple attack scenarios to demonstrate its robustness and practical viability.

II. BACKGROUND

A. *Limitations of Classical Cryptography*

In modern communication systems, cryptography plays a central role in ensuring confidentiality, integrity, and trust. Classical cryptographic methods such as RSA and Elliptic Curve Cryptography have been widely adopted because they rely on mathematical problems that are difficult for conventional computers to solve. However, the rapid progress of quantum computing has introduced new risks. Algorithms like Shor's can efficiently break the foundations of RSA and ECC, making traditional systems increasingly vulnerable. In addition to quantum threats, classical cryptography also suffers from side-channel attacks, brute-force attempts, and implementation flaws, which further weaken its long-term reliability.

To overcome these challenges, researchers have turned to quantum cryptography, which secures communication using the laws of physics rather than computational hardness. Quantum Key Distribution (QKD) protocols allow two parties to establish a shared secret key with guaranteed security. The BB84 protocol, based on photon polarization states, is valued for its simplicity and ability to detect eavesdropping through changes in the Quantum Bit Error Rate. The E91 protocol, built on quantum entanglement and Bell's inequalities, provides strong theoretical guarantees but is more complex and sensitive to noise. Together, these protocols demonstrate the potential of quantum mechanics to protect information against powerful adversaries.

Alongside quantum approaches, post-quantum cryptography (PQC) has emerged as another line of defense. PQC algorithms are designed to resist quantum attacks while running on classical hardware, making them practical for current infrastructures. Lattice-based schemes such as Kyber and NTRU are prominent candidates. Kyber, selected by NIST for standardization, offers efficient operations and strong security based on the Learning With Errors problem.

NTRU, with its long history of analysis, provides reliable encryption but involves trade-offs in key size and performance. Hash-based schemes like SPHINCS+ focus on digital signatures, offering strong security guarantees at the cost of higher computational overhead.

While quantum cryptography ensures secure key distribution, it does not provide efficient encryption. Conversely, PQC algorithms deliver robust encryption but lack the ability to detect eavesdropping during key exchange. This gap motivates the development of hybrid cryptographic systems that combine the strengths of both paradigms. By integrating BB84 for secure key generation with Kyber for efficient encryption, a balanced solution can be achieved that resists both classical and quantum adversaries. Such hybrid frameworks represent a promising direction for building next-generation secure communication systems.QKD protocols leverage principles such as the Heisenberg Uncertainty Principle and No-Cloning Theorem to detect eavesdropping.Kyber, a lattice-based algorithm, has been selected by NIST for standardization due to its efficiency and strong security guarantees.

III. PROPOSED METHODOLOGY

The proposed hybrid cryptographic system combines the BB84 Quantum Key Distribution (QKD) protocol with the Kyber Post-Quantum Cryptosystem. This integration ensures that communication remains secure against both classical and quantum attacks. The workflow includes key generation, quantum key exchange, Kyber encapsulation, hybrid key formation, encryption, decryption, and verification.

A. System Overview

The system involves two communicating entities, Alice and Bob, connected through two channels — a classical channel and a quantum channel. The classical channel handles public key exchange and message transmission, while the quantum channel is used for photon-based key generation. Both parties begin by generating their private and public keys, denoted as F_A, H_A for Alice and F_B, H_B for Bob. These public keys are exchanged through the classical channel to establish secure parameters. Once the exchange is complete, the quantum channel is activated for secure key generation.

B. Quantum Key Distribution Using BB84

In the quantum stage, Alice transmits photons encoded with random bit values using two polarization bases — rectilinear and diagonal. Bob measures the incoming photons using randomly selected bases. After transmission, both parties compare their bases over the classical channel and retain only the matching results, forming a sifted key K. The Quantum Bit Error Rate (QBER) is calculated to detect any eavesdropping attempts. If the QBER remains below the acceptable threshold, the key is considered secure. This process ensures that any interception or disturbance in the quantum channel can be detected immediately, providing real-time security assurance.

C. Kyber Post-Quantum Cryptosystem

After the BB84 key exchange, the Kyber algorithm is used to strengthen encryption. Kyber is a lattice-based key encapsulation mechanism built on the Learning With Errors (LWE) problem, which is resistant to quantum attacks. Alice performs key encapsulation using Bob's public key

H_B to generate ciphertext C_B = Encapsul(K, H_B). Bob then decapsulates the received ciphertext using his private key F_B to derive K' = Decapsul(C_B, F_B). This ensures that both parties share correlated keys derived from Kyber's lattice-based mechanism. The use of Kyber adds computational security, making the system resistant to attacks from quantum computers.

D. Hybrid Key Formation

The BB84-generated key K and the Kyber-derived key K' are combined to form a unified hybrid key

$$K_{\text{Hybrid}} = \text{Combine}(K, K'). \quad (1)$$

This hybrid key inherits the physical security of quantum key distribution and the computational strength of lattice-based encryption. The combination process ensures that even if one layer is compromised, the other remains secure. The hybrid key serves as the final encryption key for secure message transmission.

E. Encryption and Decryption Process

Using the hybrid key, Alice encrypts the message X as

$$C_X = \text{Encrypt}(X, K_{\text{Hybrid}}) \quad (2)$$

and sends it to Bob through the classical channel. Bob decrypts the ciphertext using the same hybrid key to recover the original message

$$X = \text{Decrypt}(C_X, K_{\text{Hybrid}}). \quad (3)$$

This process guarantees confidentiality, integrity, and authenticity of the transmitted data. The encryption and decryption operations are lightweight and efficient, ensuring minimal computational overhead while maintaining strong security.

F. Verification and Validation

After decryption, both parties verify that their hybrid keys are identical

$$K_{\text{HybridA}} = K_{\text{HybridB}}. \quad (4)$$

If the keys match, the communication is confirmed to be secure and free from interference. This validation step ensures synchronization and confirms that the hybrid cryptographic process has maintained integrity throughout the transmission. Any mismatch in keys indicates possible eavesdropping or transmission errors, prompting the system to discard the compromised session.

G. Attack Modeling and Security Evaluation

To evaluate robustness, several attack models are simulated, including intercept-resend, noise injection, man-in-the-middle, photon loss, and replay attacks. The system's performance is measured using parameters such as Quantum Bit Error Rate (QBER), execution time, and key consistency. The BB84 protocol demonstrates lower error rates under noisy conditions, while Kyber provides faster encryption and decryption compared to other post-quantum algorithms like NTRU and SPHINCS+. The hybrid combination of BB84 and Kyber achieves a balance between security and efficiency, making it suitable for practical deployment.

H. Summary

The proposed methodology integrates quantum and post-quantum cryptographic techniques to create a secure communication framework. BB84 ensures physical security through quantum mechanics, while Kyber provides computational resistance against quantum attacks. The hybrid

approach offers dual protection, efficient performance, and adaptability for future secure communication systems.

IV. IMPLEMENTATION

The hybrid cryptographic system was implemented in a step-by-step manner, combining the BB84 Quantum Key Distribution protocol with the Kyber Post-Quantum Cryptosystem. Each stage of the process was designed to explain the generation, exchange, combination, and validation of cryptographic key material. The mathematical expressions below describe the principal operations performed by the system. Their practical interpretation depends on the specific Python implementation and cryptographic libraries used.

A. *Environment Setup*

The implementation was carried out in a simulation environment supporting quantum-protocol modeling and classical cryptographic operations. Python was used to model random bits, basis selection, quantum-state preparation, and measurement outcomes. A Kyber-compatible cryptographic library was used for post-quantum key encapsulation. The classical channel was modeled as message exchange between Alice and Bob. This environment makes it possible to examine the protocol flow under controlled assumptions; a software simulation should not be interpreted as a physical quantum transmission unless quantum hardware or an appropriate quantum-device setup is used.

B. *Key Generation*

Both Alice and Bob generated their private and public keys:

$$(F_A, H_A) = \text{KeyGen}(),(F_B, H_B) = \text{KeyGen}() \quad (5)$$

In this expression, KeyGen() represents a key-generation procedure. The pair (F_A, H_A) denotes Alice's private and public key material, respectively, while (F_B, H_B) denotes Bob's. The subscripts identify the owner of each key pair. The private values F_A and F_B must remain confidential, whereas the public values H_A and H_B may be shared according to the protocol.

The purpose of this step is to establish the key material required for later cryptographic operations. In a Kyber implementation, the library normally generates a Kyber public/private key pair for the receiver; the exact notation used internally by the library may differ from F and H. The public key must also be authenticated when exchanged, because an unauthenticated public-key exchange can be vulnerable to a man-in-the-middle attack. This equation is a general representation of key generation and should not be interpreted as the literal internal equation of Kyber.

C. *Quantum Key Distribution (BB84)*

Alice encoded random bits into photon polarization states:

$$P = \text{polarize}(K_A, A) \quad (6)$$

Here, K_A represents Alice's random bit string, A represents the sequence of bases selected to encode those bits, and P represents the resulting encoded quantum states or their software-simulation representation. In BB84, Alice chooses each bit and its basis independently and at random. The selected basis determines how the bit is represented as a quantum state. Using two

possible bases is important because a measurement performed in the wrong basis does not, in general, reliably reveal the originally encoded bit.

This equation summarizes the encoding stage rather than specifying a particular software function. In a simulation, the oper-ation may be represented by bit-and-basis labels and rules for generating measurement outcomes. In a physical implementation, it would correspond to preparing quantum states and transmitting them through a quantum channel.

Bob measured the incoming photons:

$K_B = \text{measure}(P, B)$ (7)

In this equation, B is Bob's independently selected sequence of measurement bases, and K_B is the sequence of bits obtained from his measurements. When Bob chooses the same basis as Alice for a particular position, an ideal measurement reproduces Alice's encoded bit. When the bases differ, the measurement outcome may be random and that position is normally excluded during key sifting.

The measured sequence K_B is therefore not necessarily identical to Alice's original sequence K_A. Differences may occur because of mismatched bases, simulated or physical channel noise, or an eavesdropping attempt. The basis-comparison stage is required before the parties can identify which positions are suitable for forming a shared sifted key.

After basis comparison, the sifted key was obtained:

$$K = \text{compare}(A, B, K_A) = \text{compare}(A, B, K_B) \quad (8)$$

The function compare(·) represents the basis-sifting operation. It retains the bit positions for which Alice's basis sequence A and Bob's basis sequence B agree. For those positions, Alice uses the corresponding bits from K_A, while Bob uses the corresponding measured bits from K_B. Under ideal conditions, these retained bits match.

The equality in the expression indicates the intended result: both parties derive the same sifted key from the positions with matching bases. It does not mean that the full original bit strings are identical or that every retained bit is guaranteed to match in a noisy or attacked channel. In a practical QKD protocol, a sample of sifted bits is compared to estimate the error rate, followed by error correction and privacy amplification before the final secret key is used.

The Quantum Bit Error Rate (QBER) was calculated as:

$$\text{QBER} = \frac{\text{Number of mismatched bits}}{\text{Total bits exchanged}} \quad (9)$$

QBER measures the proportion of compared bits that differ between Alice and Bob. The numerator is the number of mismatches, and the denominator should correspond to the number of sifted bits actually compared for the estimate. If QBER is reported as a percentage, the ratio is multiplied by 100%. For example, if 5 out of 250 compared sifted bits disagree, the QBER is $(5/250) \times 100\% = 2\%$.

A low QBER indicates that the compared bit sequences agree closely, while a higher value may indicate channel noise, implementation problems, or possible eavesdropping. QBER is a useful security diagnostic, but it does not by itself prove that the channel is secure or identify the

cause of an error. The implementation should define a QBER acceptance threshold in advance. If the measured value exceeds that threshold, the candidate key should be rejected rather than used for subsequent cryptographic operations. For a complete practical BB84 system, authentication of the classical channel, error correction, and privacy amplification are also necessary.

D. Kyber Encapsulation and Decapsulation

Once the BB84 key was established and evaluated, Alice performed Kyber encapsulation using Bob's public key:

$$C_B = \text{Encapsul}(K, H_B) \quad (10)$$

This equation represents the encapsulation stage using Bob's public key H_B. The output C_B denotes the encapsulation ciphertext sent to Bob. In standard Kyber-style APIs, encapsulation usually produces both a ciphertext and a shared secret. Thus, the equation is a simplified notation: the BB84 key K is not normally the message input to Kyber encapsulation. Instead, Kyber encapsulation creates its own shared secret using the recipient's public key. If the actual implementation uses the BB84 key as an input to a separate key-derivation or combination step, that operation should be described explicitly.

The main purpose of Kyber is to establish shared secret material over a public channel using a lattice-based post-quantum mechanism. The public key can be transmitted openly, but its authenticity must be protected. The encapsulation ciphertext can also travel over the classical channel; secrecy of the resulting shared secret is provided by the cryptographic construction rather than by hiding the ciphertext.

Bob decapsulated the ciphertext using his private key:

$$K' = \text{Decapsul}(C_B, F_B) \quad (11)$$

Here, Decapsul(·) denotes the decapsulation procedure, C_B is the received Kyber ciphertext, and F_B represents Bob's private key. The operation allows Bob to recover the shared secret corresponding to the encapsulation performed by Alice. The private key must remain secret because it is used to recover the secret associated with the received ciphertext.

For a standard Kyber API, the decapsulation input is the ciphertext and the recipient's private key, and the output is the Kyber shared secret. The expected correctness condition is that this output matches the shared secret returned by Alice's encapsulation procedure. This is a functional correctness check; it does not by itself demonstrate that every possible attack has been prevented. Kyber is a key-encapsulation mechanism, so it establishes key material rather than directly encrypting arbitrary application messages.

E. Hybrid Key Formation

The BB84 key and the Kyber key were combined:

$$K_{\text{Hybrid}} = \text{Combine}(K, K') \quad (12)$$

In this expression, K represents the accepted BB84-derived key material and K' represents the Kyber shared secret. The function Combine(·) denotes the operation used to derive a common hybrid key from these two inputs. Its exact definition is important: it must be specified in the implementation rather than left as an unspecified operation.

For example, an implementation may use a documented key-derivation function to process both inputs into a key of the length required by the symmetric encryption algorithm. If direct bitwise XOR is used instead, the two inputs must have compatible lengths and the implementation must explain how they are encoded. A key-derivation function can also bind context information, such as the protocol name and session identifier, to the derived key.

The motivation for combining BB84 and Kyber is to use key material established through two different mechanisms: quantum key distribution and post-quantum key encapsulation. However, the security of the resulting hybrid key depends on the combination method, the security and authentication of each protocol, and correct implementation. The equation alone does not establish that the hybrid key automatically remains secure if either component is compromised; such a claim requires a suitable key-combination design and clearly stated assumptions.

F. Encryption and Decryption

Alice encrypted the message:

$$C_X = \text{Encrypt}(X, K_{\text{Hybrid}}) \quad (13)$$

In this expression, X denotes the original plaintext message, K_{Hybrid} is the derived key, and C_X is the ciphertext produced by the encryption algorithm. Encryption transforms the readable message into a form that should not reveal its contents to an unauthorized party who does not possess the appropriate key. The exact encryption algorithm and its required parameters must be specified in the implementation.

If AES-GCM is used, the encryption operation also generates an authentication tag and requires a nonce (initialization vector) for each encryption. The nonce does not normally need to be secret, but it must be unique for each encryption performed with the same key. The hybrid key must be converted or derived into a key length supported by the selected AES-GCM configuration. These details are needed for the encryption step to be reproducible and correctly implemented.

Bob decrypted the ciphertext:

$$X = \text{Decrypt}(C_X, K_{\text{Hybrid}}) \quad (14)$$

This equation represents the receiver's recovery of the plaintext from the received ciphertext using the corresponding key. In a more explicit notation, the recovered message may be denoted by X' and then compared with the original X. For authenticated encryption such as AES-GCM, the receiver must also verify the authentication tag; if verification fails, the plaintext must not be accepted as valid.

Successful decryption should recover the original message exactly. Confidentiality is provided

by the encryption algorithm, while integrity and authenticity of the ciphertext are provided by an authenticated-encryption mechanism when correctly configured. Therefore, these properties should not be attributed to encryption alone without specifying the algorithm and its authentication checks.

G. *Verification*

Finally, both parties verified that their hybrid keys were identical:

$$K_{\mathrm{HybridA}} = K_{\mathrm{HybridB}} \quad (15)$$

Here, K_{HybridA} is the hybrid key derived by Alice and K_{HybridB} is the corresponding key derived by Bob. Comparing these values checks whether both parties completed the key-establishment and key-combination steps consistently. In an implemen-tation, the comparison should be performed locally and should not reveal the secret key itself through the communication channel or diagnostic output.

If the equality holds, it indicates that both parties derived identical key material for that test. If the keys differ, the system should stop the session and avoid using the mismatched key. A successful equality check is evidence of functional agreement, but it is not by itself proof of overall security. The system must also validate QBER, authenticate the classical exchange, handle encryption authentication correctly, and protect private keys and session data.

The implementation therefore evaluates several distinct properties: BB84 basis sifting and QBER, Kyber shared-secret agreement, hybrid-key consistency, successful encryption and decryption, and rejection of invalid or mismatched data. These checks provide a structured way to validate the simulated workflow. Claims about execution efficiency or resistance to particular attacks should be supported by measured results from the actual implementation and by a description of the test conditions.

V. RESULTS AND ANALYSIS

This section presents the reported results for the quantum key distribution protocols, post-quantum cryptographic algorithms, and hybrid combinations. The comparisons consider QBER, execution time, throughput, memory usage, and the reported efficiency and resilience classifications.

A. *Comparison of BB84 and E91 Protocols*

Table I compares the BB84 and E91 protocols using QBER, execution time, throughput, complexity, noise sensitivity, and security strength.

TABLE I COMPARISON OF BB84 AND E91 PROTOCOLS

Metric	BB84	E91
QBER (%)	1.2	2.8
Exec. Time (ms)	12.5	15.3
Throughput (kbps)	45	38
Complexity	Low	High
Noise Sensitivity	Moderate	High
Security Strength	High	Very High

As shown in Table I, BB84 has a reported QBER of 1.2%, compared with 2.8% for E91. BB84 also has a lower execution time of 12.5 ms and higher throughput of 45 kbps, compared with 15.3 ms and 38 kbps for E91. These reported measurements indicate better efficiency for BB84 under the evaluated conditions. The table classifies E91 as having very high security strength,

although the basis for that classification should be explained in the evaluation methodology.

B. Comparison of Kyber, NTRU, and SPHINCS+

Table II compares the reported encryption time, decryption time, memory usage, throughput, and complexity of Kyber, NTRU, and SPHINCS+.

TABLE II COMPARISON OF KYBER, NTRU, AND SPHINCS+

Metric	Kyber	NTRU	SPHINCS+
Enc. Time (ms)	7.8	10.4	156
Dec. Time (ms)	7.9	9.8	149
Memory (MB)	18	22	35
Throughput (kbps)	95	70	14
Complexity	Low	Moderate	High

As reported in Table II, Kyber has the lowest encryption time (7.8 ms) and decryption time (7.9 ms), the lowest memory usage (18 MB), and the highest throughput (95 kbps) among the three entries. NTRU records 10.4 ms encryption time, 9.8 ms decryption time, 22 MB memory usage, and 70 kbps throughput. SPHINCS+ records 156 ms encryption time, 149 ms decryption time, 35 MB memory usage, and 14 kbps throughput. These comparisons indicate that Kyber has the best reported computational efficiency in this table. Ensure that the experimental method supports an encryption/decryption comparison for SPHINCS+, which is primarily a digital-signature scheme.

C. Hybrid Combinations

Table III presents the reported QBER, execution time, and efficiency classification for four hybrid combinations.

TABLE III HYBRID COMBINATIONS ANALYSIS

Combination	QBER (%)	Exec. Time (ms)	Efficiency
BB84 + NTRU	1.0	11.2	High
BB84 + Kyber	1.3	10.1	Very High
E91 + NTRU	2.5	12.6	Moderate
E91 + Kyber	2.2	12.6	High

As shown in Table III, BB84+NTRU has the lowest reported QBER (1.0%), while BB84+Kyber has the shortest reported execution time (10.1 ms) and is assigned the “Very High” efficiency classification. E91+NTRU has a QBER of 2.5%, and E91+Kyber has a QBER of 2.2%; both have a reported execution time of 12.6 ms. The results show a trade-off between QBER and execution time rather than one combination outperforming every metric.

D. Extended Comparative Metrics

Table IV compares the reported memory usage and resilience classifications of the four hybrid combinations.

TABLE IV HYBRID SYSTEMS: MEMORY USAGE AND ATTACK RESISTANCE

Combination	Memory (MB)	Resilience
BB84 + NTRU	22	High
BB84 + Kyber	18	Very High
E91 + NTRU	25	Moderate
E91 + Kyber	20	High

As presented in Table IV, BB84+Kyber has the lowest reported memory usage (18 MB) and is assigned a very high resilience classification. BB84+NTRU uses 22 MB and is classified as high; E91+NTRU uses 25 MB and is classified as moderate; and E91+Kyber uses 20 MB and is classified as high. These classifications should be supported by documented attack tests and explicit criteria in the paper.

E. Highlighting the Best Combination

The reported results indicate that BB84+Kyber has the shortest execution time in Table III and the lowest memory usage in Table IV. It is also assigned a very high efficiency and resilience classification. However, BB84+NTRU has the lowest QBER. Therefore, the preferred configuration depends on the relative importance of execution time, QBER, memory usage, and validated attack resilience.

F. Summary

The tables provide a comparison of the reported protocol and algorithm measurements. BB84 has lower QBER and higher throughput than E91 in Table I. Kyber has the best reported efficiency metrics in Table II. Among the hybrid combinations, BB84+Kyber has the shortest execution time and lowest memory usage, whereas BB84+NTRU has the lowest QBER. Further reproducible testing and clearly defined evaluation criteria are needed to validate the reported resilience classifications.

VI. CONCLUSION AND FUTURE SCOPE

The integration of BB84 with Kyber provides a secure and efficient hybrid cryptographic framework. Future work includes hardware-level optimization and large-scale deployment.

REFERENCES

[1] *C. H. Bennett and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing," in Proceedings of IEEE International Conference on Computers, Systems and Signal Processing, 1984.*

[2] *A. K. Ekert, "Quantum cryptography based on Bell's theorem," Physical Review Letters, vol. 67, no. 6, pp. 661–663, 1991.*

[3] *P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in Proceedings 35th Annual Symposium on Foundations of Computer Science, 1994.*

[4] *J. Bos et al., "CRYSTALS-Kyber: A CCA-secure module-lattice-based KEM," in IEEE European Symposium on Security and Privacy, 2018.*

[5] *V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," in Advances in Cryptology – EUROCRYPT 2010, Springer, 2010.*

[6] *J. Hoffstein, J. Pipher, and J. H. Silverman, "NTRU: A ring-based public key*

cryptosystem," in Algorithmic Number Theory, Springer, 1998.

[7] *D. J. Bernstein et al., "SPHINCS+: Submission to the NIST post-quantum project," 2019.*

[8] *H.-K. Lo, M. Curty, and K. Tamaki, "Secure quantum key distribution," Nature Photonics, vol. 8, pp. 595–604, 2014.*

[9] *N. Gisin et al., "Quantum cryptography," Reviews of Modern Physics, vol. 74, pp. 145–195, 2002.*

[10] *M. Mosca, "Cybersecurity in an era with quantum computers: Will we be ready?" IEEE Security & Privacy, vol. 16, no. 5, pp. 38–41, 2018.*

[11] *C. Peikert, "A decade of lattice cryptography," Foundations and Trends in Theoretical Computer Science, vol. 10, no. 4, pp. 283–424, 2016.*

[12] *R. Renner, "Security of quantum key distribution," International Journal of Quantum Information, vol. 6, no. 1, pp. 1–127, 2008.*

[13] *S. Pirandola et al., "Advances in quantum cryptography," Advances in Optics and Photonics, vol. 12, no. 4, pp. 1012–1236, 2020.*

[14] *National Institute of Standards and Technology (NIST), "Post-Quantum Cryptography Standardization," 2022.*

[15] *M. Amiri and R. Safavi-Naini, "Hybrid post-quantum and classical key exchange protocols," in Proceedings of ACM CCS Workshop on Post-Quantum Cryptography, 2017.*

[16] *T. Sasaki et al., "Practical security evaluation of BB84 and E91 protocols under realistic noise," IEEE Transactions on Quantum Engineering, vol. 2,*

pp. 1–12, 2021.

[17] *K. Zeng, "Physical layer key generation in wireless networks: Challenges and opportunities," IEEE Communications Magazine, vol. 53, no. 6, pp. 33–39, 2015.*

[18] *A. Broadbent and C. Schaffner, "Quantum cryptography beyond quantum key distribution," Designs, Codes and Cryptography, vol. 78, pp. 351–382, 2016.*

[19] *M. Nielsen and I. Chuang, Quantum Computation and Quantum Information, Cambridge University Press, 2010.*

[20] *S. Aaronson, "The limits of quantum computers," Scientific American, vol. 298, no. 3, pp. 62–69, 2008.*