IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67

A Comprehensive Study of Crystal Kyber

Kulsum Abdullah Sayed, Nabila Anwar Jamal Qureshi, Mishkat Moinuddin Khan, Shifa Farooqui
Computer Engineering
M. H. Saboo Siddik College of Engineering, Byculla, Mumbai, India
kulsum.222267.co@mhssce.ac.in, nabila.222265.co@mhssce.ac.in
Indiamishkat.222264.co@mhssce.ac.in, shifa.211217.co@mhssce.ac.in

Abstract: *Quantum computing threatens classical cryptographic systems, prompting the need for quantum-resistant algorithms. This paper introduces Crystal Kyber, a lattice-based Key Encapsulation Mechanism (KEM) standardized by the National Institute of Standards and Technology (NIST) as a post-quantum cryptographic solution. KEMs enable secure key exchange over insecure channels, facilitating encryption and authentication. Crystal Kyber is based on the Module Learning with Errors (MLWE) problem, a hard mathematical problem resistant to both classical and quantum attacks. The mechanism securely encapsulates a shared secret key, ensuring that only the intended recipient can decapsulate it. Crystal Kyber offers three parameter sets—Kyber512, Kyber768, and Kyber1024—balancing security and performance for various applications, from constrained environments to high-security domains. This paper examines Crystal Kyber's key generation, encapsulation, and decapsulation processes, highlighting its computational efficiency, scalability, and quantum resistance. It positions Crystal Kyber as a crucial component of future-proof cryptographic standards for securing communication in the quantum era..*

Keywords: Advanced Encryption Standard, Federal Information Processing Standard, Key-Encapsulation Mechanism, Learning with Errors, MLWE-Module Learning with Errors, NIST- National Institute of Standards and Technology, NISTIR-NIST Interagency or Internal Report, NTT-Number-Theoretic Transform, PKE-Public-KeyEncryption, PQC-Post-Quantum Cryptography, PRF-Pseudorandom Function, RBG-RandomBit Generator, SHA-Secure Hash Algorithm

I. INTRODUCTION

The rapid advancements in quantum computing present a significant threat to the security of classical public-key cryptosystems such as Rivest–Shamir–Adleman (RSA) and Elliptic Curve Cryptography (ECC). These systems rely on the computational difficulty of mathematical problems like integer factorization and discrete logarithms, which are considered infeasible for classical computers to solve within a reasonable time. However, with the advent of quantum computers, algorithms like Shor's algorithm could efficiently solve these problems, potentially rendering current cryptographic methods obsolete. This poses a severe risk to the confidentiality and integrity of internet communications, financial transactions, and sensitive government data.

To address these challenges, the field of post-quantum cryptography has gained considerable attention, focusing on developing cryptographic protocols that remain secure even against quantum adversaries. Among the leading approaches is lattice-based cryptography, which is based on hard mathematical problems such as the Learning With Errors (LWE) problem and its more advanced variant, the Module Learning With Errors (MLWE) problem. These problems are believed to be resistant to quantum attacks due to their computational complexity.

The Module Lattice-Based Key Encapsulation Mechanism (ML-KEM), derived from CRYSTALS-KYBER—a NISTselected post-quantum cryptographic algorithm—leverages the hardness of the MLWE problem to enable secure key exchanges. ML-KEM facilitates the secure establishment of shared secret keys over public channels, ensuring data security in the post-quantum era. The security of ML-KEM makes it a viable candidate for protecting communications in a world where quantum computers could compromise existing systems.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67

This research project focuses on the design, implementation, and performance evaluation of ML-KEM to establish its suitability for real-world post-quantum cryptographic infrastructures. The primary objectives of this project are as follows:

- Design and implementation: Develop the core components of ML-KEM, including key generation, encapsulation, and decapsulation procedures, adhering to the specifications of the MLWE problem
- Secure Key Exchange: Enable the secure exchange of keys for symmetric encryption and authentication over public networks, ensuring communication remains confidential and tamper-proof.
- Security and Performance Evaluation: Analyze MLKEM's performance across three parameter sets: MLKEM-512, ML-KEM-768, and ML-KEM-1024. Each parameter set offers a different balance of security and efficiency, catering to various use cases and threat models.
- Quantum Resistance: Validate ML-KEM's resistance to quantum attacks by conducting a thorough theoretical security analysis based on the IND-CCA2 (Indistinguishability under adaptive Chosen-Ciphertext Attack) security model.
- Post-Quantum Readiness: Demonstrate ML-KEM's practicality and readiness for deployment in post-quantum cryptographic systems, highlighting its potential as a secure and efficient key encapsulation mechanism.

II. LITERATURE SURVEY

A. Overview of Existing Solutions

Before the advent of quantum-resistant cryptography, data security heavily relied on classical public-key cryptographic systems like RSA, Elliptic Curve Cryptography (ECC), and Diffie-Hellman. These systems depend on the difficulty of solving mathematical problems such as integer factorization and the discrete logarithm, which are computationally infeasible for classical computers to solve within a reasonable time. RSA, for example, secures data by using a public and private key pair, with security ensured by the challenge of factoring large prime numbers. Similarly, ECC provides encryption based on the difficulty of solving discrete logarithms in elliptic curve groups. However, these cryptographic systems are vulnerable to quantum attacks. Quantum algorithms like Shor's algorithm can efficiently solve both integer factorization and discrete logarithms, which means that a sufficiently powerful quantum computer could break RSA,ECC, and DiffieHellman encryption. This looming threat makes traditional crypto graphic methods ineffective in the face of quantum computing advancements, highlighting the urgent need for quantum-resistant encryption algorithms that can secure data against both classical and quantum attacks.

B. Comparative Study of Various Approaches

This comparative table below outlines different approaches for post-quantum cryptography:

- FIPS 203- ML-KEM Standard: Focuses on a theoretical lattice-based encryption mechanism (ML-KEM) with a primary emphasis on protection against quantum attacks. It is a secure but complex approach lacking extensive real-world implementation.
- Lattice-Based Encryption Using ElGamal: Simpler and based on the SIS problem,this method is easier to implement but needs more scalability and guidance for broader use.
- CRYSTALS-Kyber Implementations: Both the Portable and Masked versions of CRYSTALS-Kyber offer strong security with optimizations for efficiency. The masked implementation adds protection against side-channel attacks but still lacks thorough real-world application strategies.

C. Algorithms

In this project, various algorithms plays a pivotal role in the system's overall effectiveness. Several algorithms are wellsuited for the task of communication from one party to another.

There is a list of algorithms provided by FIPS 203. We have included all the algorithms, categorizing them based on the section they fit into best or the underlying concept they operate on. Algorithm 1 and Algorithm 2 are example algorithms explained for simpler functions; hence, we have only listed them without summarizing their operation.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67

Conversion and Compression Algorithms: The conversion and compression algorithms are essential for transforming data between different representations, ensuring efficient storage and computation in cryptographic processes.

- Algorithm 1 ForExample()
- Algorithm 2 SHAKE128example()
- Algorithm 3 BitsToBytes() - This algorithm converts a sequence of bits into an array of bytes, where every 8 bits form one byte in little-endian order. It is used in encoding binary data for structured storage and transmission.
- Algorithm 4 BytesToBits() - The reverse of BitsToBytes, this algorithm extracts individual bits from a byte array, reconstructing the original bit sequence. It is useful in cryptographic applications requiring precise bit-level manipulation.
- Algorithm 5 ByteEncode() - This algorithm compresses an array of integers (such as polynomial coefficients) into a more compact byte representation. It reduces the size of transmitted or stored data, optimizing cryptographic computations.
- Algorithm 6 ByteDecode() - The inverse of ByteEncode, this algorithm reconstructs integer values from a compressed byte array. It is used when retrieving polynomial coefficients or other numerical data from encoded storage.

Sampling Algorithms: Sampling algorithms generate random polynomials necessary for secure cryptographic computations, ensuring unpredictability in key generation and encryption.

- Algorithm 7 SampleNTT() - This algorithm takes a random seed and two indexing bytes as input to generate a polynomial in the Number-Theoretic Transform (NTT) domain. The use of NTT helps in efficient polynomial multiplication, a key operation in lattice-based cryptography.
- Algorithm 8 SamplePolyCBD()- It generates noise polynomials with coefficients sampled from a centered binomial distribution (CBD). These noise polynomials are crucial for the security of ML-KEM, as they make attacks on the underlying lattice problem computationally infeasible.

Number-Theoretic Transform (NTT) Algorithms: These algorithms facilitate efficient polynomial multiplication, which is fundamental to the cryptographic operations of ML-KEM.

- Algorithm 9 NTT() - The Number-Theoretic Transform (NTT) converts a polynomial from its coefficient representation to a special frequency-domain format, where multiplication is performed more efficiently. This is analogous to the Fast Fourier Transform (FFT) used in signal processing.
- Algorithm 10 NTT-1() - This algorithm performs the inverse of NTT, converting a polynomial back from the frequency domain to its coefficient representation. It ensures that transformed polynomials can be correctly interpreted after computation.
- Algorithm 11 MultiplyNTTs() - This algorithm efficiently multiplies two polynomials in the NTT domain. Since multiplication in the frequency domain is much faster than in the coefficient domain, this significantly speeds up cryptographic operations.
- Algorithm 12 BaseCaseMultiply() - Used as the base case in recursive polynomial multiplication, this algorithm handles small-degree polynomials directly, ensuring efficient multiplication when recursion reaches its limit.

Key-Encapsulation Mechanism (KEM) Algorithms These algorithms form the foundation of the ML-KEM cryptosystem, enabling secure key encapsulation and decapsulation.

- Algorithm 13 K-PKE.KeyGen() - This algorithm generates a public-private key pair for the K-PKE encryption scheme, which is an internal component of ML-KEM. The public key is used for encryption, while the private key is used for decryption.
- Algorithm 14 K-PKE.Encrypt() - This algorithm encrypts a plaintext message using the public key. It ensures that only the holder of the corresponding private key can decrypt and retrieve the original message.
- Algorithm 15 K-PKE.Decrypt() - The decryption algorithm uses the private key to recover the plaintext from a given ciphertext. It verifies integrity and ensures the confidentiality of the transmitted message.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67

ML-KEM Algorithms: These are the main cryptographic operations in ML-KEM, responsible for key generation, encapsulation, and decapsulation.

- Algorithm 16 ML-KEM.KeyGen-internal() - This internal algorithm generates encapsulation and decapsulation key pairs within the ML-KEM framework. It ensures the keys are correctly structured for use in later steps.
- Algorithm 17 ML-KEM.Encaps-internal() - This algorithm encapsulates a shared secret key within an encrypted message. It helps establish a secure communication channel between two parties.
- Algorithm 18 ML-KEM.Decaps-internal() - This algorithm decapsulates a received ciphertext to recover the original shared secret. It verifies the authenticity of the encapsulated data before decrypting it.
- Algorithm 19 ML-KEM.KeyGen() - The main key generation algorithm in ML-KEM, it produces an encapsulation key (public) and a decapsulation key (private) used for secure key exchange.
- Algorithm 20 ML-KEM.Encaps() - This algorithm generates a shared secret key and encrypts it using the encapsulation key. The resulting ciphertext is sent to the receiver, who can decrypt it using the decapsulation key. • Algorithm 21 ML-KEM.Decaps() - The final step in the ML-KEM process, this algorithm decrypts the ciphertext using the decapsulation key to retrieve the shared secret. If the ciphertext is valid, both sender and receiver will have the same secret key for secure communication

III. METHODOLOGY

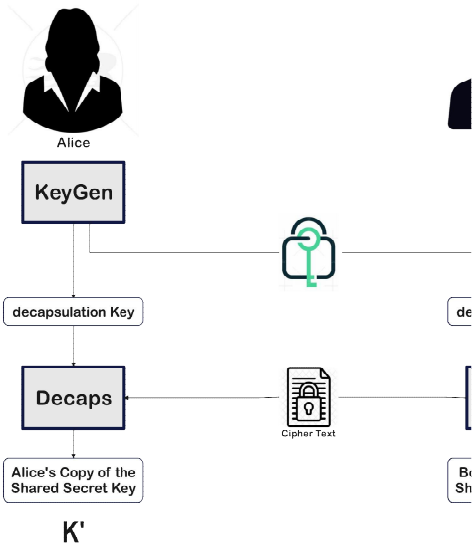

Fig. 1. System's Architecture.

The above diagram Fig 1 represents a post-quantum key exchange protocol based on the Kyber algorithm. The overall process involves two parties, Alice and Bob, securely exchanging a shared secret key over an insecure channel using the properties of lattice based problems, specifically the Modular Learning With Errors (MLWE) problem. Here's a breakdown of the steps involved in the system architecture:

A. Key Generation (KeyGen)

Alice begins by generating a key pair by following the steps below:

- She creates a random matrix A and computes a secret vector s and error/noise vector e.
- These are typically performed using NTT Polynomial operations (Number Theoretic Transform), which is efficient in lattice-based cryptography.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67

- Alice's key generation step outputs a decapsulation key (used for decrypting later) and her public key, which is sent to Bob.

B. Encapsulation (Encaps)

Bob receives Alice's public key and uses it in the encapsulation process:–

- He solves the MLWE problem by performing matrix multiplication with Alice's public key to generate a shared secret.
- Bob calculates the ciphertext (cipher text) by encrypting his copy of the secret key using Alice's public key. 3) Polynomial arithmetic in Rq (a polynomial ring) ensures the computations are performed efficiently under modular arithmetic.
- The encapsulated result (ciphertext and decapsulation key) is sent back to Alice.

C. Decapsulation (Decaps)

Alice performs the decapsulation operation:

1) Using her private key, she performs inverse polynomial operations and decodes the ciphertext to recover the shared secret key.

2) The MLWE-based decryption process allows her to extract her own copy of the shared secret key K' from the data sent by Bob.

D. Shared Key (K, K')

Both Alice and Bob now have their respective copies of the shared secret key. Bob's shared key is denoted as K, and Alice's shared key is denoted as K'.

The key exchange is successful when K = K', meaning both Alice and Bob hold the same shared secret key, which can be used for further cryptographic purposes, like encrypting communication.

The Ultimate Framework and the Design Process

The process of our project is outlined below, including the type of algorithm used at each step. This has been studied and presented to you in the form of a diagram. Each step involves a specific algorithm to be performed. A comprehensive list of all the algorithms can be found in Section C of the literature review. We now proceed to discuss the three major steps in detail.

Key Generation.

Key Encapsulation. • Key Decapsulation.

Key Generation.

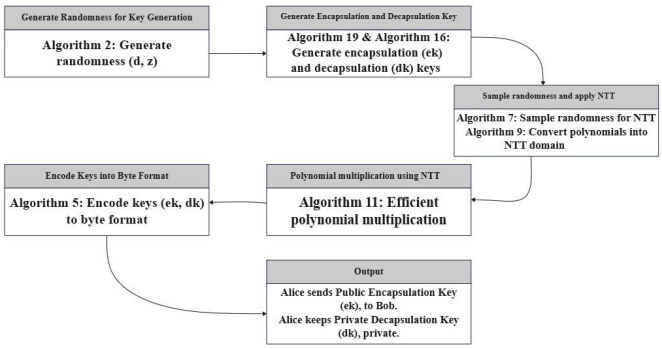

Fig. 2. Flow of Key Generation

In the Fig 2 Key Generation phase (on Alice's side), the process begins by generating randomness using SHAKE128 (Algorithm 2), which produces random values required for key creation. These random values are then used in ML-KEM.KeyGen (Algorithm 19), with an internal call to ML-KEM. KeyGen-internal (Algorithm 16), to generate the

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67

public encapsulation key (ek) and private decapsulation key(dk). Alice also samples randomness and applies the Number Theoretic Transform (NTT) using SampleNTT (Algorithm 7) and NTT (Algorithm 9) to transform polynomials into the NTT domain for efficient computation. The keys are encoded into byte format using ByteEncode (Algorithm 5) to be securely transmitted. Alice keeps her private decapsulation key (dk) and sends the public encapsulation key (ek) to Bob.

Key Encapsulation.

In the Encapsulation phase (on Bob's side) illustrated above in the Fig 3, Bob first generates randomness for encapsulation using SHAKE128 (Algorithm 2) and uses Alice's public key (ek) to encapsulate a message. He applies ML-KEM. Encaps (Algorithm 20), with an internal call to ML-KEM. Encaps-internal (Algorithm 17), to produce the ciphertext (c) and the shared secret key (K). Bob again samples randomness and applies NTT using SampleNTT (Algorithm 7) and NTT (Algorithm 9), which allows for efficient polynomial multiplication using MultiplyNTTs (Algorithm 11).

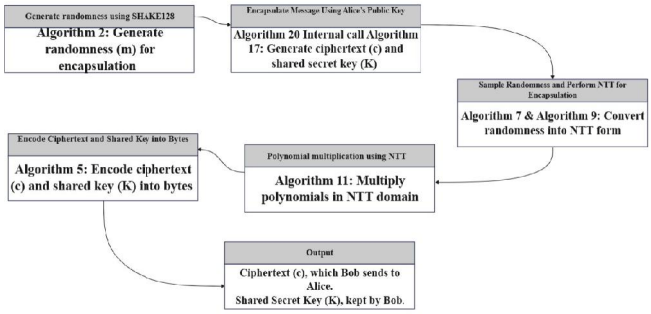

Fig. 3. Flow of Key Encapsulation.

The ciphertext and shared key are then encoded into byte format using ByteEncode(Algorithm 5). Bob sends the ciphertext to Alice and retains the shared key (K).

Key Decapsulation.

In the above Fig 4 Finally, in the Decapsulation phase (on Alice's side), Alice receives the ciphertext from Bob and decodes it using ByteDecode (Algorithm 6). She samples randomness again and convert the ciphertext back

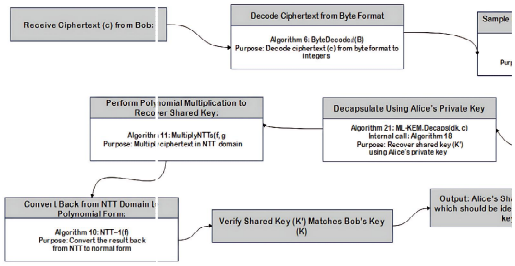

Fig. 4. Flow of Key Decapsulation.

into the NTT domain using SampleNTT(Algorithm 7) and NTT(Algorithm 9). Alice performs polynomial multiplication with her private key using MultiplyNTTs (Algorithm 11) to recover the shared key. She converts the result back to normal form from the NTTdomainusingNTT¹(Algorithm10). By invoking ML-KEM.Decaps (Algorithm 21) and its internal call to ML-KEM.Decaps internal (Algorithm 18), Alice successfully decapsulates the ciphertext and retrieves the shared secret key (K). Finally, Alice verifies that her shared key matches Bob's, allowing them to securely communicate.

DOI: 10.48175/IJARSCT-25711

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67

Empirical Analysis

This Table. 1 presents a comparative analysis of different cryptographic approaches, focusing on their security foundations, key algorithms, and practical implementation aspects. Table I summarizes the characteristics of Post-Quantum Cryptography, the FIPS 203 ML-KEM standard, Lattice-Based Encryption using ElGamal, and masked implementations of CRYSTALS-Kyber.

This comparative analysis highlights the trade-offs between different cryptographic approaches in terms of security, efficiency, and practical deployment. While ML-KEM and lattice-based cryptography provide strong post-quantum security guarantees, implementations such as CRYSTALSKyber with SIMD optimizations enhance performance for real-world applications. Additionally, masked implementations address security concerns related to side-channel attacks, ensuring robustness against advanced adversaries. These insights can guide the selection of cryptographic schemes based on application-specific requirements.

TABLE I. COMPARATATIVE ANALYSIS.

Aspect	Post-Quantum Cryptography Overview	FIPS 203 - ML-KEM Standard	Lattice-Based Encryption Using ElGamal	Portable Efficient CRYSTALS-Kyber Implementation (WAsm)	Masked Implementations of CRYSTALS Kyber
Key Algorithms Discussed	Codebase, multivariate, hash-based,isogenybased cryptography	Module-Lattice-Based Key-Encapsulation Mechanism (ML-KEM)	Short Integer Solution (SIS)-based encryption	CRYSTALS-Kyber using SIMD opti-mizations	CRYSTALS-Kyber with ”Double and Check” and Look-Up-Table (LUT) masking methods
Security Focus	Protects against quantum threats by developing new algorithms	Based on noisy linear equations which are hard for quantum computers	Enhances security Usinga lattice-basedSIS problem	Strong security with efficiency forweb and IoT applications	Protects against side channel attacks with masked implement-ations
Practical Implementation	Limited Focuson practical useand developer guidance	Primarily theoretical; Lacksrealworld implementation strategies	Simple to implement but lacks scalability guidance	Optimized for web environments but needs diverse platform testing	Focuses on masked techniques; lacks com-prehensive real-world application

IV. CONCLUSION

The NIST FIPS 203 (DRAFT) outlines the design of the Module-Lattice-Based Key Encapsulation Mechanism (MLKEM), a post-quantum cryptographic standard for secure key exchange in the era of quantum computing. It uses

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67

three parameter sets corresponding to different security strengths: ML-KEM-512 (Category 1), ML-KEM-768 (Category 3, recommended default), and ML-KEM-1024 (Category 5). These parameter sets balance security and performance, affecting key sizes, ciphertext sizes, and decapsulation failure rates. MLKEM relies on the Number Theoretic Transform (NTT) for efficient polynomial multiplication, reducing time complexity from O(n2) to O(nlogn), and avoids floating-point arithmetic to ensure security. Its deterministic core functions, Encaps and Decaps, ensure reliable shared secret generation and retrieval.

This project, implementing the Kyber algorithm, lays the groundwork for further research in post-quantum cryptography. Future directions include broad adoption in industries such as finance, healthcare, and telecommunications, the development of hybrid cryptosystems combining Kyber with classical algorithms, and optimization for IoT and edge devices. Additionally, work on improving performance, contributing to open-source libraries, and ensuring compliance with international standards is essential. Kyber also has potential applications in securing blockchain transactions and digital identities.

The implementation of ML-KEM is expected to significantly advance the security of cryptographic systems, particularly in the context of post-quantum cryptography. It provides a robust framework for quantum-resistant key encapsulation, ensuring the safe exchange of cryptographic keys even against potential quantum computing threats. Key anticipated outcomes include establishing quantum-resistant cryptography standards to replace vulnerable traditional systems, encouraging the adoption of standardized cryptographic practices, and ensuring high reliability with minimal decapsulation failure rates. Furthermore, the standard supports future algorithmic improvements to adapt to the evolving landscape of quantum computing, fostering continuous advancements in security and performance.

In conclusion, ML-KEM and the Kyber algorithm offer a robust solution for securing communications against quantum threats. As quantum computing evolves, continued research and development in post-quantum cryptography will be crucial to ensure the longevity and security of cryptographic systems.

REFERENCES

[1]. J. Bos, L. Ducas, E. Kiltz, et al., ”CRYSTALS–Kyber: A CCAsecure module-lattice-based KEM,” in *Proc. IEEE Euro SP*, 2018, pp. 353–367.
[2]. R. Avanzi et al., ”CRYSTALS-Kyber Algorithm Specifications and Supporting Documentation,” NIST PQC Project, 2022. [Online]. Available: https://pq-crystals.org/kyber
[3]. D. J. Bernstein, ”Post-quantum cryptography,” *Nature*, vol. 549, pp. 188–194, 2017.
[4]. D. Liu, F. Zhang, Y. Wang, and J. Wang, ”Hardware Implementation of CRYSTALS-Kyber on FPGA,” *IEEE Access*, vol. 9, pp. 14515–14525, 2021.
[5]. M. Mosca, ”Cybersecurity in an Era with Quantum Computers: Will We
[6]. Be Ready?” *IEEE Security Privacy*, vol. 16, no. 5, pp. 38–41, 2018.
[7]. P. W. Shor, ”Algorithms for Quantum Computation: Discrete Logarithms and Factoring,” in *Proc. 35th Annual Symposium on Foundations of Computer Science*, 1994, pp. 124–134.
[8]. L. K. Grover, ”A fast quantum mechanical algorithm for database search,” in *Proc. 28th ACM STOC*, 1996, pp. 212–219.
[9]. Gheorghiu, T. Kapourniotis, and E. Kashefi, ”Quantum Cryptography: A Survey,” *ACM Computing Surveys*, vol. 51, no. 6, pp. 1–36, 2019. [9] J. Daemen and V. Rijmen, ”AES Proposal: Rijndael,” NIST, 1999.
[10]. Chen and W. Liu, ”The Impact of Grover’s Algorithm on AES Key Length,” in *Proc. IEEE TrustCom*, 2017, pp. 1093–1096.
[11]. M. Grassl et al., ”Applying Grover’s Algorithm to AES: Quantum Resource Estimates,” in *Post-Quantum Cryptography*, Springer, 2016, pp. 29–43.
[12]. M. Roetteler, M. Naehrig, K. M. Svore, and Q. Wang, ”Quantum Resource Estimates for Computing AES Key Search,” *IACR Cryptology ePrint Archive*, 2017.
[13]. S. Gueron and N. Mouha, ”Hybrid Post-Quantum Key Encapsulation Mechanism (HPKE) with Kyber,” IETF Draft, 2023. [Online]. Available:

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67

[14]. https://datatracker.ietf.org/doc/draft-irtf-cfrg-hpke/
[15]. J. Bernstein, T. Lange, and C. van Vredendaal, ”Post-Quantum Cryptography: State of the Art,” in *Handbook of Blockchain, Digital Finance, and Inclusion*, vol. 2, 2018, pp. 343–384
[16]. NIST, ”Post-Quantum Cryptography Standardization: Round 3 Submissions,” 2020. [Online]. Available: https://csrc.nist.gov/Projects/postquantum-cryptography/round-3-submissions
[17]. T. Oder, T. Schneider, T. Poppelmann, and T. G¨ uneysu, ”Practical CCA2-¨ Secure and Masked Ring-LWE Implementation,” in *Proc. CHES*, 2018.
[18]. Peikert, ”A Decade of Lattice Cryptography,” *Foundations and Trends in Theoretical Computer Science*, vol. 10, no. 4, pp. 283–424, 2016.
[19]. Y. Liu, Y. Liu, and Y. Zhang, ”A Performance Evaluation of Kyber and Other Lattice-Based KEMs,” in *IEEE ICC Workshops*, 2022.
[20]. Aggarwal et al., ”Quantum Attacks on Public-Key Cryptosystems,” *IACR Cryptology ePrint Archive*, 2017.
[21]. R. Perlner and D. Cooper, ”Quantum Resistant Public Key Cryptography: A Survey,” NIST IR 8105, 2016.
[22]. A Childs and W. van Dam, ”Quantum Algorithms for Algebraic Problems,” *Reviews of Modern Physics*, vol. 82, no. 1, pp. 1–52, 2010.
[23]. S. Alani, ”Quantum Computing and its Impact on Cryptography,” *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 2, 2019.
[24]. A Basso, G. Costantino, and L. Catuogno, ”On the Post-Quantum Security of AES,” *Journal of Information Security and Applications*, vol. 67, 2022.
[25]. M. Amy, O. Di Matteo, and M. Mosca, ”Estimating the Cost of Generic Quantum Pre-image Attacks on SHA-2 and SHA-3,” *IACR Transactions on Symmetric Cryptology*, vol. 2017, no. 1.
[26]. A Hulsing et al., ”On the Quantum Security of Hash-Based Signatures,”¨ in *Post-Quantum Cryptography*, Springer, 2016.
[27]. Alkim, L. Ducas, T. Poppelmann, and P. Schwabe, ”Post-quantum¨ Key Exchange—A New Hope,” in *Proc. USENIX Security*, 2016, pp. 327–343.
[28]. A Gentry, ”Fully Homomorphic Encryption Using Ideal Lattices,” in *STOC*, 2009, pp. 169–178.
[29]. A Stebila and M. Mosca, ”Post-quantum Key Exchange for the Internet and the Open Quantum Safe Project,” in *Symposium on Security and Privacy Workshops (SPW)*, 2016
[30]. A H. Bennett and G. Brassard, ”Quantum Cryptography: Public Key Distribution and Coin Tossing,” in *Proc. IEEE International Conference on Computers, Systems and Signal Processing*, 1984.
[31]. T. Prest, ”Learning with Errors and Ring-LWE,” in *Mathematics of Modern Cryptography*, Springer, 2021.