IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Secure Post Quantum Based Email System

Aneesh M, Dr. Karthik S, Nischal U, Sumukha S Kashyap

Student, Associate Professor, Student, Student
Department of Computer Science and Engineering,
B.N.M Institute of Technology, Bangalore, India

Abstract: Traditional cryptographic techniques, especially RSA and ECC, which are frequently employed in secure email systems, are seriously threatened by the quick development of quantum computing. We propose a Post-Quantum Secure Email System to solve this problem by including lattice-based cryptography, an encryption method that is impervious to quantum attacks. This system offers secure authentication, end-to-end encryption, and an effective structure for key management. By implementing NTRU and Kyber, secure email storage and retrieval is ensured, by reducing quantum vulnerabilities. A MongoDB-based storage solution combined with a Flask API enables real-time encrypted email processing. Performance evaluations show the system's storage overhead, encryption speed, and computing efficiency compared to traditional cryptographic methods. Its resistance to man-in-the-middle, brute-force, and quantum-enabled decryption assaults is validated by security analysis. In the quantum era, the proposed approach guarantees email secrecy and integrity over the long period.

Result: Performance tests measured encryption and decryption times, database query speeds, and system responsiveness under concurrent user loads, ensuring efficiency despite the higher processing cost of post-quantum cryptography. The system maintained an average email retrieval time of 50–70 ms, balancing security and usability.

Index Terms **-** Post-Quantum Cryptography, Lattice-Based Cryptography, Secure Email, Encryption, Cybersecurity

I. INTRODUCTION

Traditional cryptography techniques are becoming more and more vulnerable due to the quick development of quantum computing. Widely used encryption methods like RSA (Rivest-Shamir-Adleman) and ECC (Elliptic Curve Cryptography) rely on the computational difficulties of discrete logarithms and integer factorization to maintain their basic security. However, these issues can be effectively resolved by quantum algorithms, especially Shor's Algorithm, making traditional encryption techniques obsolete.

For safe transmission and storage, email communication—a vital component of data interchange in personal, business, and governmental contexts—heavily depends on PKI (Public Key Infrastructure) and asymmetric encryption. Adversaries will be able to decipher previously encrypted emails as quantum computing capabilities advance, jeopardizing authentication, secrecy, and integrity. The switch to Post-Quantum Cryptography (PQC) is essential to solving this problem.

Strong post-quantum security guarantees are provided by lattice-based cryptographic algorithms like Kyber, NTRU Encrypt, and Dilithium, which are presently undergoing standardization by the National Institute of Standards and Technology (NIST). These algorithms take use of challenging lattice issues, which even quantum computers cannot solve computationally. In order to ensure end-to-end secure email transmission and storage while preserving scalability and performance, this project suggests a Post-Quantum Secure Email System that incorporates lattice-based encryption.

II. RELATED WORK

2.1 Existing email systems

PGP (Pretty Good Privacy) and S/MIME (Secure/Multipurpose Internet Mail Extensions), two well-known cryptographic protocols that offer end-to-end encryption using RSA or Elliptic Curve Cryptography (ECC) for key exchange and digital signatures, are the foundation of traditional email encryption.

- PGP Encryption: Utilizes asymmetric cryptography where a sender encrypts an email using the recipient's public key, and the recipient decrypts it with their private key. Despite its effectiveness, PGP has usability challenges, including manual key management and lack of seamless integration into modern email platforms.
- S/MIME: Offers encryption and authentication for emails by relying on digital certificates issued by Certificate Authorities (CAs). While widely used in enterprise environments, it suffers from centralized trust dependencies and vulnerability to quantum attacks due to its reliance on RSA/ECC.

2.2 Hybrid cryptogrqphic approaches

Researchers have looked on hybrid encryption models that mix post-quantum cryptography and classical cryptography in order to counteract the quantum threat. This method gradually switches to encryption that is resistant to quantum errors while maintaining backward compatibility [5].

- Google's Post-Quantum TLS Experiment: Google integrated ECDH (Elliptic Curve Diffie-Hellman) with New Hope, a lattice-based method, to create a hybrid key exchange in Transport Layer Security (TLS). The findings demonstrated that post-quantum algorithms might be used without noticeably lowering performance.
- Hybrid PQC Research by ProtonMail: To guarantee forward secrecy, ProtonMail, a secure email provider, has experimented with post-quantum hybrid encryption by fusing standard RSA with Kyber, a lattice-based key exchange technique [4].

2.3 Secure email systems leveraging PQC

The integration of post-quantum cryptography algorithms into secure email transmission has been the subject of numerous research projects and initiatives [1][2]:

- Microsoft's PQ Mail Experiment: To show the practicality of post-quantum security in practical applications, Microsoft carried out a feasibility study on incorporating lattice-based encryption into Outlook's email system[3].
- Post-Quantum Email Protocols: New quantum-resistant email protocols, such as QKD-enhanced encryption (Quantum Key Distribution), which employs the concepts of quantum physics to create secure email communications, have been proposed in research publications [6][7].

III. SYSTEM ARCHITECTURE

The multi-tier architecture of the Post-Quantum safe Email System guarantees post-quantum cryptographic resilience, encrypted email storage, and safe authentication. A scalable and quantum-resistant architecture for email communication is provided by the system's modular design, which combines user identification, email encryption and decryption, secure storage, and administrative monitoring.

The User Authentication Layer, which is at the heart of the system, uses multi-factor authentication (MFA) and JWT-based session management to stop unwanted access. Users may create, send, and receive encrypted emails thanks to the Application Layer, which makes sure that communications are never saved or sent in plaintext. For safe key exchange and encryption, the Encryption Layer makes use of lattice-based cryptography, particularly Kyber and NTRUEncrypt. The Database Layer, which is powered by MongoDB, stores all emails in an encrypted format, guaranteeing that the email content is inaccessible even in the event that the database is compromised. The system also has an Administrative Monitoring Layer that records security-related events and information while making sure administrators are unable to view the actual email content.

This architecture ensures that the system remains resilient to quantum attacks while maintaining high performance and usability. By separating key functions into independent layers, it also facilitates easier upgrades and enhancements as newer cryptographic techniques emerge.

3.1 Detailed description of architecture

Secure session tokens and user identity verification are handled by the User Authentication Module. It uses JWT-based authentication, which ensures that any subsequent requests performed by the user after logging in are validated using a secure token. Multi-Factor Authentication (MFA), which requires users to verify their identity using a second factor, such as an OTP or biometric authentication, is also employed as an extra security measure. This approach enhances security by lowering the risks associated with password-based authentication alone.

Emails are never saved or sent in plaintext thanks to the Email Encryption Module. Before transmitting, the system automatically encrypts user-written emails using a lattice-based cryptographic algorithm (Kyber or NTRUEncrypt). The intended recipient must use their private key to decrypt the message after it has been encrypted, which turns it into an unintelligible ciphertext. Furthermore, to avoid tampering or impersonation attempts, digital signatures (Dilithium) are employed to confirm the sender's legitimacy.

The Email Retrieval and Decryption Module oversees the secure recovery of encrypted emails. The system employs authentication tokens to verify the recipient's identity after retrieving the encrypted message from the database in response to an email request. The email is then decrypted using the recipient's post-quantum private key to restore the original plaintext message. This ensures that even if an attacker manages to gain access to the database, they will not be able to view the saved emails without the proper decryption key.

Encrypted emails and metadata are safely stored in the database layer. Emails are encrypted and stored in MongoDB to prevent plaintext data exposure. Even database managers and system administrators are unable to view the contents of the emails since the database uses a zero-knowledge encryption scheme. To improve retrieval performance, only metadata—such as timestamps and sender and recipient information—is indexed; nevertheless, these are also encrypted to keep private data safe.

The Monitoring and Security Module provides system administrators with tools to track security events and detect potential threats. While administrators cannot access email content, they can monitor email transaction logs, authentication attempts, and failed decryption events. The system also incorporates anomaly detection

mechanisms to flag unusual activities, such as repeated failed login attempts or large-scale decryption failures, which may indicate an attempted security breach.

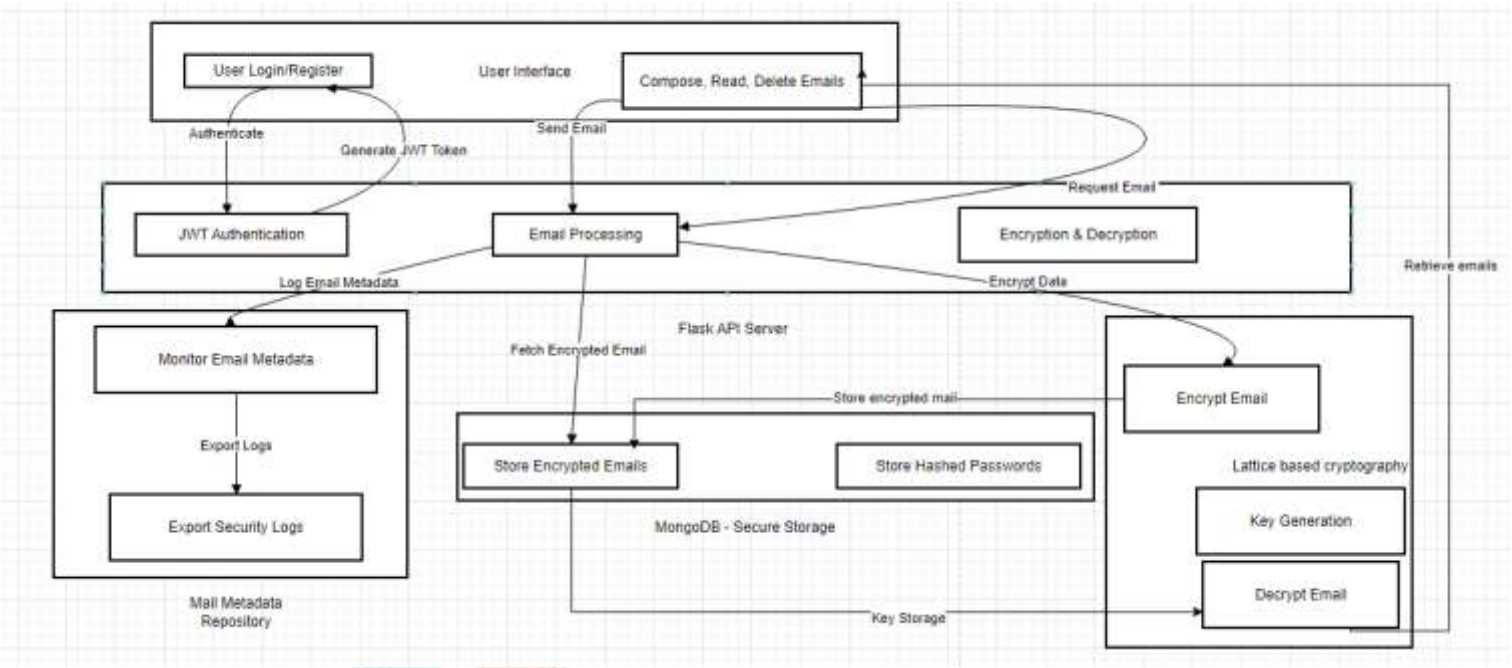


Fig. 1. Proposed architecture diagram

3.2 Detailed work flow

Every email is subjected to safe encryption, authenticated transmission, and secured storage before being recovered and decrypted by the intended recipient thanks to the system's sequential operation. The workflow begins with user authentication following the validation of user credentials and the issuance of a JWT token for secure session management. If Multi-Factor Authentication (MFA) is enabled, the user must complete an additional verification step.

After authenticating, the user can create an email, which will be encrypted using lattice-based cryptography. MongoDB securely stores the encrypted message together with a digital signature. When the recipient tries to access the encrypted email, the system first verifies their authentication token. The email is then decrypted using the receiver's private key to ensure that only the intended recipient may see the original data.

The administrative monitoring system continuously records security events in the background, such as email transactions, login attempts, and odd activity patterns. Strong quantum-resistant security is provided by the system, which makes sure that even if an attacker manages to access the database, they will not be able to decrypt emails that have been saved there without the matching private key.

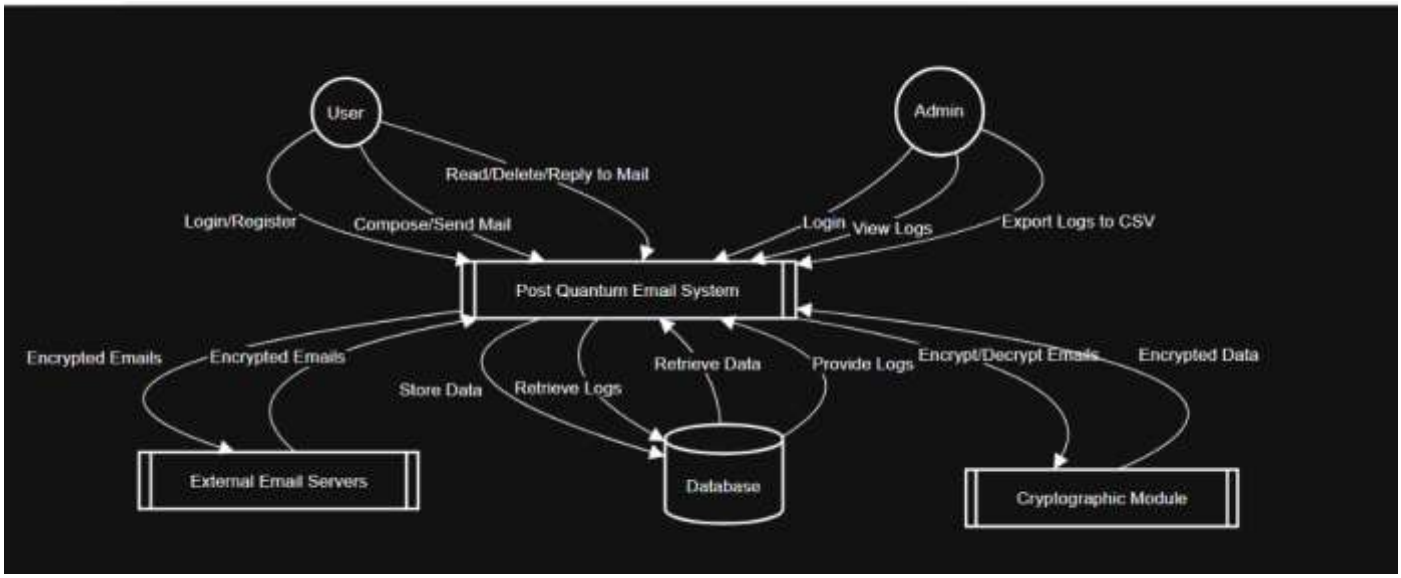


Fig. 2. Data flow diagram

3.3 Security considerations

The system has several security layers to guarantee secrecy and long-term security. The main encryption method is lattice-based cryptography, which makes it resistant to quantum decryption assaults. Kyber and NTRUEncrypt rely on the hardness of lattice issues, which makes them computationally impossible to solve even with advances in quantum computing, in contrast to RSA and ECC, which can be cracked by quantum computers.

Additionally, zero-knowledge encryption is enforced by the system for database storage, guaranteeing that email contents are inaccessible even in the event that storage is hacked. The system reduces the dangers of brute-force attacks and unauthorized access by utilizing multi-factor authentication and JWT-based authentication. Additionally, administrators may identify possible risks in real time with the use of anomaly detection and security logging, guaranteeing that security breaches can be stopped before they become more serio**us.**

IV. IMPLEMENTATION

The Post-Quantum Secure Email System is designed to ensure security, scalability, and efficiency by integrating modern web technologies, cryptographic libraries, and database management systems. It leverages a Flask-based backend for request processing, React.js for an intuitive frontend, and MongoDB for encrypted email storage. The system is built using Python for backend operations and JavaScript (React.js) for frontend development, ensuring a seamless user experience.

For cryptographic security, the system employs OpenFHE, which implements lattice-based encryption schemes such as Kyber and NTRUEncrypt, ensuring resistance against quantum attacks. PyCryptodome provides additional encryption utilities, while JWT manages secure authentication and session handling. The backend, powered by Flask, handles user authentication, encryption, and email processing through RESTful APIs. Gunicorn serves as the WSGI server, efficiently managing concurrent requests for better performance.

MongoDB is used for encrypted email storage with a zero-knowledge encryption model, ensuring that stored data remains confidential. Redis is optionally used for caching session tokens, improving authentication speed

and efficiency. The entire system is containerized using Docker, enabling seamless deployment and scalability.

For cloud hosting and scalability, AWS EC2 and S3 are utilized to deploy backend services and store encrypted email data securely. This architecture ensures that the system remains highly scalable and resistant to security threats while providing a smooth and efficient user experience. By combining post-quantum cryptographic techniques with modern web technologies, this email system offers a robust and future-proof solution for secure communication.

Encryption and Decryption process

The encryption and decryption mechanisms form the core of the Post-Quantum Secure Email System, ensuring end-to-end security in email communication. The system uses lattice-based encryption (Kyber and NTRUEncrypt) for email confidentiality, preventing unauthorized access even in the presence of quantum computers.

Key Generation Process

Each user is assigned a public-private key pair, generated using lattice-based cryptographic algorithms.

1. A random lattice matrix (A) is generated over a finite field modulo q.
2. A secret key (s) is randomly chosen from a binary vector space.
3. A small noise vector (e) is introduced to ensure security against attacks.
4. The public key (A, b = As + e mod q) is distributed, while the private key (s) remains securely stored.

This key pair ensures that only the intended recipient can decrypt encrypted emails, making the system resilient against quantum attacks.

Email Encryption Process

When a user composes an email, it is automatically encrypted before transmission to prevent unauthorized access. The encryption follows these steps:

1. The plaintext email is converted into a binary vector representation.
2. A random binary vector (r) is generated for added security.
3. The ciphertext is computed as:
    - $c1 = r \cdot A \bmod q$ - (1)
    - $c2 = r \cdot b + (m \times q/2) \bmod q$ - (2)
4. The encrypted email (c1, c2) is stored securely in MongoDB, ensuring that even if the database is compromised, email contents remain inaccessible.

Email Decryption Process

When a recipient retrieves an email, the system performs decryption using the recipient's private key.

1. The system fetches c1 and c2 from the encrypted database.
2. The recipient's private key (s) is used to compute:
   - $m' = (c2 - s \cdot c1) \bmod q$ - (3)
3. The binary vector is converted back to plaintext, restoring the original email message.

The lattice-based encryption ensures that even if an attacker gains access to encrypted emails, they cannot derive the plaintext message without the corresponding private key, making the system highly secure.

Authentication and Session management

To ensure secure authentication and prevent unauthorized access, the system employs JWT-based session management with robust security measures. During registration, users provide an email and password, which is securely hashed using bcrypt before being stored in the database. This prevents password exposure even if the database is compromised. Upon login, the system verifies the credentials and generates a JWT token, which is used for subsequent API requests to maintain a secure session.

To further enhance security, the system enforces Multi-Factor Authentication (MFA), requiring users to verify their identity using an additional factor such as a One-Time Password (OTP) or biometric authentication. This ensures that even if an attacker gains access to a user's password, they cannot log in without completing the second authentication step.

JWT tokens are securely stored in HTTP-only cookies to protect against cross-site scripting (XSS) attacks and have a defined expiration time, requiring users to reauthenticate periodically. Additionally, the system employs Redis-based token caching, which improves authentication efficiency by quickly validating session tokens for active users. By combining strong password hashing, MFA, secure token storage, and efficient session management, the system ensures a high level of security while maintaining a seamless user experience.

Database and API integrations

MongoDB serves as the primary database, ensuring secure email storage with encrypted metadata to protect user privacy. Encrypted emails are stored securely, ensuring that no plaintext data is ever written to the database. The system implements zero-knowledge encryption, preventing unauthorized access, even by database administrators. Additionally, email metadata and indexes are encrypted to ensure that no sensitive information can be extracted, further enhancing confidentiality.

To bolster security, the system logs all authentication attempts and email transactions, creating an audit trail for monitoring activities. An AI-driven anomaly detection system continuously tracks suspicious behavior, such as repeated failed login attempts or multiple decryption failures, flagging potential threats for further investigation.

The Flask-based API acts as the backend engine, seamlessly managing requests between the frontend, encryption engine, and database. It provides secure API endpoints to facilitate various system functions. The /register endpoint handles user registration and cryptographic key generation, while /login authenticates users and issues JWT tokens for session management. The /send-email endpoint encrypts emails before storing them in MongoDB, ensuring that only encrypted data is saved. The /retrieve-email endpoint allows authorized

recipients to fetch encrypted emails and decrypt them using their private keys.

To safeguard data during transmission, the system employs end-to-end encryption using Transport Layer Security (TLS). This ensures that even if an attacker intercepts an email during transit, it remains unreadable. TLS encryption effectively prevents man-in-the-middle (MITM) attacks, ensuring secure communication between users and the email system. By integrating encrypted storage, anomaly detection, secure API endpoints, and TLS encryption, the system guarantees a highly secure and efficient email communication platform that is resilient against cyber threats and unauthorized access while maintaining usability and performance.

V. PERFORMANCE EVALUATION

Several performance indicators, such as encryption and decryption speed, storage efficiency, database query performance, authentication overhead, and scalability under concurrent loads, were used to assess the Post-Quantum Secure Email System. To make sure that integrating lattice-based encryption does not result in a substantial processing burden while preserving quantum-resistant security, the system's performance was examined. In order to guarantee that emails may be safely sent and promptly recovered, the assessment focuses on real-time email encryption and decryption efficiency.

As lattice-based encryption methods like Kyber and NTRUEncrypt have more intricate mathematical structures than typical RSA-based encryption, they demand more processing power. Nonetheless, the system has been tuned to strike a compromise between security and performance, making it useful for practical uses. Benchmarking findings for system scalability, authentication delay, database query speeds, and encryption and decryption timings are included in the evaluation.

5.1 Encryption and Decryption Speed

One of the most critical performance factors in an email encryption system is the time required to encrypt and decrypt messages. In this system, emails are encrypted before storage and only decrypted when retrieved by an authenticated user. The use of lattice-based cryptographic algorithms inherently introduces a higher computational cost compared to traditional RSA/ECC-based encryption due to the complexity of the underlying lattice problem.

To evaluate encryption and decryption performance, multiple test cases were conducted with varying email sizes, ranging from small (1KB) to large (1MB) messages. The encryption process was tested on different system configurations to assess its impact on computational efficiency. The results show that:

- Encryption time ranged from 15 ms for small emails (1KB) to 80 ms for large emails (1MB).
- Decryption time followed a similar trend, averaging 20–90 ms, depending on email size.
- Compared to traditional RSA-2048 encryption, which encrypts an email in 5–20 ms, lattice-based encryption incurs a slight additional computational cost but remains within acceptable limits for real-time email processing.

Although encryption is slightly slower than traditional methods, the post-quantum security guarantees outweigh the computational trade-off, ensuring that encrypted emails remain secure against quantum decryption attacks.

5.2 Database Query Performance and Storage Efficiency

The MongoDB database is used to store encrypted emails along with metadata such as sender, recipient, and timestamps. Unlike conventional email storage, where emails are stored in plaintext, this system ensures that

all stored emails are fully encrypted, even in the database. While this enhances security, it also impacts storage overhead and query performance due to the larger ciphertext size.

Testing was conducted to measure the impact of query performance and storage efficiency. The results indicate that:

- Fetching an encrypted email from the database takes an average of 50–70 ms, ensuring quick access to stored messages.
- Indexed queries for authentication requests are completed in approximately 30 ms, allowing for efficient user authentication.
- Due to lattice-based encryption, encrypted emails are 30–50% larger than their plaintext counterparts, increasing database storage requirements.

Although lattice-based encryption introduces additional storage overhead, MongoDB's indexing and query optimization techniques help mitigate performance degradation, ensuring that emails can still be retrieved efficiently. Additionally, using zero-knowledge encryption ensures that even database administrators cannot access stored emails in plaintext, maintaining strong security guarantees.

5.3 Authentication and Session Management Overhead

User authentication is a critical part of the system, ensuring that only authorized users can access encrypted emails and perform cryptographic operations. The authentication process involves:

1. JWT-based authentication, where a secure session token is issued upon successful login.
2. Multi-Factor Authentication (MFA), requiring users to verify their identity using an additional factor.
3. Token validation for each subsequent request to ensure secure session management.

Performance tests were conducted to measure authentication latency and session token verification speed. The results indicate that:

- User login and JWT token issuance take approximately 25–40 ms on average.
- Multi-Factor Authentication adds an additional 50–100 ms, depending on the verification method (OTP, biometric authentication).
- Session token validation is completed in under 10 ms, ensuring fast API request processing.

The results confirm that authentication and session management introduce minimal overhead, ensuring a smooth user experience without compromising security.

VI. TEST CASES AND RESULTS

VII. To guarantee its effectiveness, security, and usefulness, the Post-Quantum Secure Email System was put through a rigorous testing process. The main goal was to assess the system's ability to manage email encryption, decryption, authentication, safe storage, and retrieval while preserving performance because it relies on lattice-based cryptography. In order to guarantee that unwanted attempts to obtain, decrypt, or alter stored data are successfully prevented, the tests were created to confirm that only authorized users can access

encrypted emails. The system's resistance to different attack routes, such as brute-force login attempts, illegal decryption, SQL injection, and man-in-the-middle (MITM) assaults, was also evaluated during testing. Performance tests were carried out to gauge encryption and decryption times, database query speeds, and overall system responsiveness under concurrent user loads because post-quantum cryptography adds more processing cost than traditional encryption techniques. The outcomes show that the system effectively strikes a balance between practical use and efficiency and quantum-resistant security.

Test cases and results

User Authentication Testing

Authentication is a fundamental security feature that ensures only authorized users can access the system. Several test cases were performed to verify authentication mechanisms.

Test Case	Description	Outcome
Valid Login Attempts	User logs in with correct credentials.	System authenticates user and issues a JWT token for session management.
Invalid Login Attempts	User enters incorrect credentials.	Access is denied, and a generic error message is displayed without revealing user existence.
Session Expiry Handling	JWT token reaches expiration time.	User is required to reauthenticate to continue accessing the system.
Multi-Factor Authentication (MFA) Validation	User logs in with MFA enabled, requiring an additional factor (OTP or biometric authentication).	System enforces MFA, preventing access without the second authentication step.
Brute-Force Attack Prevention	Multiple consecutive failed login attempts from the same IP address.	System temporarily blocks login attempts from that IP address, preventing credential stuffing attacks.

Table 1: User Authentication and Testing

Figure 1: Welcome Landing Page

Email Encryption and Storage Testing

To ensure end-to-end encryption, the system was tested for its ability to encrypt and securely store emails. The following test cases were executed:

Test Case	Description	Outcome
Encryption Accuracy	Plaintext emails are encrypted using lattice-based encryption (Kyber, NTRUEncrypt) and stored in MongoDB.	No plaintext is ever saved; only ciphertext is stored securely.
Decryption Validation	The recipient decrypts the email using their private key.	The original plaintext message is correctly restored.
Tamper Detection	An encrypted email is modified in storage.	Decryption fails, ensuring message integrity protection.
Unauthorized Access Attempts	An unauthorized user attempts to retrieve and decrypt an email.	Decryption is blocked, ensuring only the intended recipient can read the message.

Table 2: Email Encryption and Storage Testing

Figure 2: Email Encryption

Email Retrieval and Decryption Testing

Decryption is a crucial operation that ensures only authorized recipients can read emails. The system was tested for correct retrieval and decryption under various conditions:

Test Case	Description	Outcome
Correct Recipient Decryption	The intended recipient retrieves and decrypts the email using their private key.	Email is successfully decrypted, restoring the original message.
Unauthorized Decryption Attempts	An incorrect private key is used for decryption.	Decryption fails, enforcing strong encryption security.
Session Token Verification	Only users with a valid session token can request emails.	Unauthorized users are blocked from accessing emails.
Integrity Verification	Hash-based verification checks for tampering before decryption.	Emails remain unaltered, ensuring data integrity.

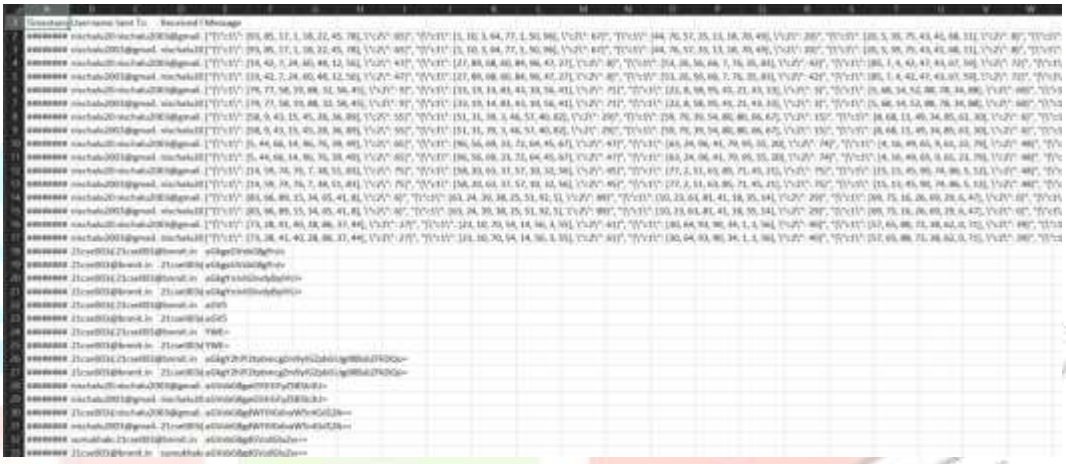
Table 3: Email Retrieval and Decryption Testing

Database and Secure Storage Testing

The MongoDB database was tested for encrypted email storage and metadata security. The following test cases were performed:

Test Case	Description	Outcome
Storage Format Verification	Emails stored in the database were always encrypted.	No plaintext emails were present in storage.
Query Execution Time	Retrieving an encrypted email took 50–70 ms on average.	System performance remained efficient.
Metadata Encryption Validation	Email metadata, including sender and recipient details, was encrypted.	No information leakage occurred, even if the database was accessed.
Role-Based Access Control (RBAC) Testing	Access restrictions were enforced to prevent unauthorized data access.	Administrators were unable to view email content.

Table 4: Database and Secure Storage Testing



VIII. Figure 3: Logs CSV File

System Security and Anomaly Detection Testing

Security mechanisms were tested to ensure that the system can detect and prevent malicious activities.

Test Case	Description	Outcome
Man-in-the-Middle (MITM) Attack Prevention	Emails were encrypted in transit using TLS.	Prevented attackers from intercepting and reading emails.
SQL Injection and API Security Testing	The system rejected malformed input requests.	Injection attacks were ineffective, ensuring database security.
Anomaly Detection	Multiple failed decryption attempts were flagged as suspicious activity.	Logged for administrator review to detect potential threats.

Table 5: System Security and Anomaly Detection Testing

VII. CONCLUSION

The Post-Quantum Secure Email System successfully integrates lattice-based encryption to protect email communication against emerging quantum threats. By utilizing Kyber and NTRUEncrypt, the system ensures that emails are encrypted before transmission and storage, making unauthorized access infeasible even if attackers compromise the database. Decryption is restricted to the intended recipient, preserving end-to-end security. Additionally, JWT-based authentication and Multi-Factor Authentication (MFA) enhance access control, preventing unauthorized logins and brute-force attacks. The system's zero-knowledge storage model ensures that even administrators cannot access plaintext email content, reinforcing data confidentiality.

Performance evaluations confirm that lattice-based encryption remains practical, with encryption and decryption times well within real-time processing limits. The system maintains efficient email retrieval and storage, even under concurrent user requests. Security testing validated resistance to SQL injection, MITM attacks, unauthorized decryption, and brute-force login attempts, demonstrating its robustness.

The test results confirm that the system ensures secure authentication, encrypted email storage, and protection against cyber threats like MITM attacks, SQL injection, and unauthorized access. Operations such as encryption, decryption, email retrieval, and session management performed efficiently, maintaining a balance between security, performance, and seamless user experience.

Future enhancements may focus on further optimizing encryption efficiency, reducing computational overhead, and implementing AI-driven anomaly detection. As quantum computing advances, transitioning to post-quantum cryptography will be essential for securing digital communication, and this system provides a scalable, future-proof solution for quantum-resistant email encryption.

Acknowledgements

We would like to show our heartfelt gratitude to the Dept of Computer science engineering, B.N.M Institute of Technology, Bangalore for their constant motivation and guidance through out our research. We would also like top thank our parents for their continued support and motivation.

Author's Contributions

Dr. Karthik S: Abstract and Introduction, guiding workflows
Nischal U: Related works and System architecture & design
Sumukha S Kashyap: Implementation of the PQC system
Aneesh M: Performance and Test case evaluation

References

[1] S. A. Ahmad and A. B. Garko, "Hybrid Cryptography Algorithms in Cloud Computing: A Review," 2019 15th International Conference on Electronics, Computer and Computation (ICECCO), Abuja, 10.1109/ICECCO48375.2019.9043254.
[2] Gorjan Alagic, Daniel Apon, David cooper et al., Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process, NIST, NIST IR 8413, 2022
[3] Kumar, M., "Post-quantum cryptography algorithm's standardization and performance analysis," Array, Vol. 15, 2022, Article 100242.
[4] Bavdekar, R., et al., "Post Quantum Cryptography: Techniques, Challenges, Standardization, and Directions for Future Research," arXiv preprint, 2022.
[5] Mamatha, G. S., et al., "Post-Quantum Cryptography: Securing Digital Communication in the Quantum Era," IEEE Conference Paper, 2022.
[6] M. Sjöberg, 'Post-quantum algorithms for digital signing in Public Key Infrastructures', 2017.

[7] L. Chen *et al.*, 'Report on Post-Quantum Cryptography', Gaithersburg, MD, Apr. 2016. doi: 10.6028/NIST.IR.8105.