QUANTUM-RESISTANT ENCRYPTION FOR SECURE END-TO-END COMMUNICATION

Sameer, Kant[1]; Neha Kishor, Jadhav[1]; Jawar, Singh[1]; Dilip, Singh[2]; Anjan Kumar, Singh[3]

[1]Department of Electrical Engineering, Indian Institute of Technology Patna, Patna, Bihar, INDIA
[2]Department of Telecommunications, Government of India Office of Additional DG Telecom, MPLSA-Bhopal, India
[3]Ministry of Electronics and Information Technology, India

ABSTRACT

Cryptographically Relevant Quantum Computers (CRQCs) are no longer hypothetical; they present significant challenges to current IT infrastructure by potentially breaking existing encryption schemes within minutes. This paper introduces an innovative and efficient method for achieving quantum-resistant encryption through lattice-based cryptography. We specifically tackle the challenge of encrypting extremely small units of data, such as a single letter or a single-bit message, by constructing a multidimensional lattice. Our proposed technique leverages the Short Vector Problem (SVP) in lattice-based cryptography and incorporates the Learning with Errors (LWE) methodology for data encryption and decryption. We demonstrate the feasibility and robustness of this approach using a real-time messaging application that provides quantum-resistant end-to-end encryption. Our work has the potential for deployment in strategic applications, securing information from the "harvest now, decrypt later" threat, even in the presence of quantum technologies.

Keywords - Quantum Cryptography, Lattice-based, Learning with errors, Short vector Problem, Quantum-Resistant, Public-key encryption

1. INTRODUCTION

In today's rapidly evolving digital landscape, the security of sensitive information faces unprecedented threats, especially with the advent of quantum computing. Traditional cryptographic algorithms, such as the widely used RSA (Rivest-Shamir-Adleman) public key encryption, are encountering formidable challenges. The emergence of quantum computing, with its unparalleled computational power, poses a significant risk to the reliability of these conventional encryption schemes. Cryptographically Relevant Quantum Computers (CRQCs) can effortlessly break existing algorithms, thereby undermining data security. In response to this urgent need, the scientific community initially focused on lattice cryptography because cryptographic constructions based on lattice designs came with security proofs derived from worst-case lattice problems. Ajtai and Dwork [8] proposed the first lattice-based encryption scheme, which was later refined and streamlined by introducing the concept of the Learning With Errors (LWE) problem. The LWE problem, an intermediate problem asymptotically at least as hard as some common worst-case lattice problems, proved relatively easy to use in cryptographic constructions, representing a significant advancement [1].

1.1 Related Work

The National Institute of Standards and Technology (NIST) has approved four quantum-resistant algorithms. Notably, Kyber [9], a Key Encapsulation Mechanism (KEM) based on lattice-based cryptography and the Modular Learning with Errors (MLWE) problem, targets the GapSVP problem [1]. Other lattice-based algorithms include Crystal-Dilithium [21], SABER [15], and NTRU [20]. Kyber is preferred for its efficiency in secure communication and resistance to quantum attacks, thanks to its use of Number Theoretic Transforms (NTT) and noise during ciphertext generation, which complicates attacks like BKZ [19], LLL reduction, BKW, and primal attacks [16].
Oded Regev's work [17] was pivotal in proving the quantum hardness of the Learning With Errors (LWE) problem and Shortest Vector Problem (SVP), underpinning the security of these lattice-based methods. Efforts to update the Signal protocol [13] and the Messaging Layer Security (MLS) protocol developed by the Internet Engineering Task Force (IETF) [13] for end-to-end secure messaging are ongoing but not yet completed.

1.2 Our Contribution

In this work, we introduce a lattice-based encryption algorithm designed to resist quantum attacks, inspired by the Learning With Errors (LWE) scheme and the Shortest Vector Problem (SVP). By integrating Oded Regev's encryption methods with techniques from Kyber, we developed a new ciphertext generation approach. Our algorithm optimizes public key storage by generating the key matrix from a seed and two vectors, reducing storage and transmission overhead while enhancing security through increased ciphertext noise. We have implemented this algorithm in a server-client model for end-to-end quantum-resistant communication. The messaging application we are developing utilizes a CCA-secure public key encryption (PKE) scheme. Unlike the typical approach of exchanging a symmetric key via a quantum-safe Key Encapsulation Mechanism (KEM) like Kyber, and then encrypting the message, our method directly encrypts the message using public key cryptography. This

approach simplifies key management, eliminates the need for an additional symmetric key exchange, and reduces potential vulnerabilities, making it suitable for applications where overhead and complexity are critical concerns.
Our algorithm has demonstrated strong performance in both encryption and decryption. We've focused on reducing ciphertext size and optimizing multiplication complexity, aiming for a time complexity of $O(n \log n)$ rather than $O(n^2)$. This ensures both efficiency and security in establishing a highly secure communication channel.
Lastly, we present use cases for our messaging solution, showing its applicability in secure governmental communications, financial transactions, and private messaging, where quantum-resistant security is essential.

2. PRELIMINARIES

2.1 Notations

The set of integers is denoted by $\mathbb{Z}, \mathbb{Z}_{q}$ represents the finite field of cardinality q and its elements are represented by the integers in the interval $[-[\frac{q}{2}], \frac{q}{2}]$. $B(n)$ denotes an encoded array of n integers like $\{a_1, a_2, a_3...., a_n\}$ each integer takes 4 bytes each makes the value range of 0 to $2^{32}-1$. $\mathcal{D}$ denotes the Gaussian distribution and $\mathcal{D}_\sigma$ with standard deviation σ. Here, $x \leftarrow \chi$ denotes the set of x belongs from the distributed error terms generated which is denoted by χ. $\Pr(X)$ denotes the Probability of X.B_n^k denotes an array of 32 bits representing the ASCII value for a character.
Extendable Output Function (XOF) is a cryptographic primitive that takes a seed as input and produces an output of any desired length. It's like a versatile cryptographic tool that can generate data according to specified distributions. Let's say we have a seed *seed* and we want to use XOF to create matrices and vectors. We start by initializing XOF with the seed seed, and then we use it to generate the matrices and vectors [22]. For example, we can generate a matrix A by calling XOF with a specific seed seed_A, and similarly, we can generate a vector b by calling XOF with a seed seed_b.
Let seed be the seed used by the XOF, a variants of SHA (secure hashing algorithm)-256 and SHA-128 pseudo number generators in these functions which give us a hash value of a definite length. To generate a matrix A, we call XOF with seed seed_A:

$$A \leftarrow \text{XOF } (\text{seed}_A)$$

Similarly, to generate a vector b, we call XOF with seed seed_b:

$$b \leftarrow \text{XOF } (\text{seed}_b)$$

The process is deterministic, meaning that for a given seed, XOF will always produce the same output. However, in practice, XOF may use random oracles to achieve this determinism while statistically approximating the desired distribution.
The G function that when we send the $A.s + e$ and s_t combine the function output the results as stored in p_t as $\left(\sum_{i=0}^{256} A \cdot (s_i - s_{s_{t_i}})\right)$ so that the redundant lattice dimensions which was chosen randomly from 256 dimensions will not the be added in the resultant p_t which form the public key.
H function is used in the decryption algorithm where it takes the ciphertext and uses it to provide the absolute difference between the ciphertext matrix value and the value that came from putting s in those equations in ciphertext with redundant dimensions and then we check the probability that it is less than $\frac{q}{4}$ or not for decoding the bits.

2.2 Gaussian Distribution

As Regev [1] demonstrates a quantum reduction from worst-case lattice issues to LWE with Gaussian error, given arbitrarily many LWE samples, the Gaussian distribution is the default error distribution for classical LWE.
Definition. Let $\mathcal{D}_\sigma$ be the Gaussian distribution with standard deviation σ. The rounded Gaussian distribution function $\mathcal{D}_\sigma^{\mathcal{R}}$ is defined as follows:
For any $x \in \mathbb{R}$, the sample y from $\mathcal{D}_\sigma^{\mathcal{R}}$ is obtained by rounding x to the nearest integer and adding a sample from $\mathcal{D}_\sigma$.
In mathematical notation:

$$y = \lfloor x + z + 0.5 \rceil$$

where z is a sample from $\mathcal{D}_\sigma$. If X follows a Gaussian distribution then the rounded Gaussian of X_Z will be

$$\Pr(X_{\mathbb{Z}} = N) = \int_{N+1/2}^{N-1/2} g_{\mu,\sigma}\, dx$$

In the following $\mu = 0$.
Goal. In here we have to find the distribution of $X_{\mathbb{Z}_q} = X_{\mathbb{Z}}$ mod q.

$$\Pr(X_{\mathbb{Z}_q} = [a]) = \Pr(X_{\mathbb{Z}} = a + kq) \quad \text{for some } k \in \mathbb{Z}$$

$$\sum_{k \in \mathbb{Z}} \int_{a+kq+1/2}^{a+kq-1/2} g_{0,\sigma}(x)\, dx = \int_{a+1/2}^{a-1/2} f_{\sigma,q}(x)\, dx$$

Where

$$f_{\sigma,q}(x) : x \to \sum_{k \in \mathbb{Z}} \frac{1}{\sigma\sqrt{2\pi}} \exp \frac{-(x+kq)^2}{2\sigma^2}$$

2.3 LWE definition

Learning With Errors (LWE) is a fundamental problem in lattice-based cryptography and serves as the basis for many encryption schemes and cryptographic protocols[1][3][4]. It involves the challenge of distinguishing between noisy information and random noise in a set of equations. terms e_i.
Notation. Let χ be the rounded gaussian mod q and $e_i \leftarrow \chi$.
Definition. $q > 0$ be a modulus and $m, n > 0$. Now given m samples of such equations of n dimesnions in the form of $\mathbf{a_i}, < \mathbf{a_i}, \mathbf{s_i} > +\mathbf{e_i}$ where $\mathbf{a_i}$ is uniform at random in $\mathbb{Z}_q^n$ and $e_i \leftarrow \chi$. Find the secret $s \in \mathbb{Z}_q^n$

The LWE can be formulated more compactly as a matrix-vector equation:

$$X \cdot s + e = k \mod p$$

where:

- X is a known matrix with rows corresponding to coefficients x_i.
- e is a vector of error terms e_i.
- k is a vector of known values k_i.

The LWE problem is challenging because the error terms e_i are typically drawn from a random distribution, making it difficult to distinguish between the true solution and random noise. The LWE's security relies on the hardness of solving this problem, even when given multiple noisy equations. In practical lattice-based cryptography, LWE is used to build secure encryption schemes such as Ring-LWE, which is a variant of LWE, and many other cryptographic primitives. The security of these schemes is based on the assumption that solving the LWE problem is computationally infeasible, even for powerful adversaries.

Hardness. The LWE problem is difficult for several reasons. First of all, even quantum algorithms don't seem to be able to help because the most effective LWE algorithms operate in exponential time. Second, since it is a logical extension of the Learning Parity with Noise (LPN) issue, it has been well explored and is generally accepted to be a challenging topic in learning theory. Furthermore, as LPN involves the issue of decoding from random linear binary codes, any advancement in algorithmic techniques for LPN is likely to result in advances in coding theory. It is known that LWE is hard because of several assumptions about the worst-case difficulty of typical lattice problems like SIVP (the shortest independent vectors problem) and GapSVP (the decision version of the shortest vector problem) [1][4]. More specifically, the conventional assumption that GapSVP is difficult to estimate within polynomial factors underlies difficulties when the modulus q is exponential. The hardness is based on somewhat less common (but still quite credible) assumptions for polynomial moduli q, which is the more important setting for cryptographic applications. To be more precise, this means that either GapSVP or SIVP are difficult to approximate within polynomial factors, even with a quantum hint, or that GapSVP is difficult to estimate even when provided with a short basis [3].

3. ALGORITHM AND SPECIFICATIONS

The algorithm is based on the LWE encryption scheme which was introduced by Odded Regev. We adopted the approach [10] for the generation of the public matrix A and used the SVP problem to make the public key and secret key. The main difference in creating the lattice dimensions in our algorithm and others is that we are using the concept of reduced dimensions which means we would not take 'k' dimensions in counter to calculate the resultant lattice equations and that would work as our second secret key. It also increases the complexity of lattice reduction techniques. Similar to other post-quantum cryptography (PQC) algorithms in which parameters play an important role in their security strength. We have also defined a set of parameters in it. The algorithm is parameterized by n, m, k, q, γ.

n	m	k	γ	q
256	150	25	5	3907

- The parameter n is set to 256 to utilize a plaintext size of 256 bits, ensuring 256 bits of entropy for key encapsulation. A lower n reduces noise and affects security.
- The parameter q is chosen to ensure a negligible probability of decryption failure, similar to Kyber, where a prime number around 2^{12} defines a good security parameter.
- The parameter m represents the number of LWE samples chosen to resist BKW and BKZ attacks with LLL reduction techniques.
- The parameter γ balances security and computational timing, determining the number of equations selected from the public key for a single bit with modulo q.
- The parameter k is the number of equations chosen from m samples in the public key matrix. A subset of k rows, when combined, produces different equations to encrypt a single bit.

3.1 Key Generation

At the beginning, several arrays (sk, sk_t, pk, pk_t, a) and a prime number (prime) are initialized. These arrays are used to store secret keys, public keys, and other parameters used in the encryption and decryption processes. The "generateUniqueRandomNumbers" function is defined to generate a specified number of unique random numbers within a given range. This function is used to generate secret keys, public keys, and other random values required for encryption by SHAKE-128 in its standard variant and AES-256 in CTR mode in its 90s variant as XOFs to generate pseudorandom output for cryptographic operations. The secret_key_gen and public_key_gen functions are defined to generate secret keys (sk, sk_t) and public keys (pk, pk_t) based on random numbers generated using the XOF.

Algorithm 1 KeyGen(): Key generation

$\sigma, \rho \leftarrow \{0,1\}^{256}$ and $\alpha \leftarrow \{0,1\}^{k}$
$a \sim B(n) :=$ XOF(σ)
$p \sim C(k) :=$ XOF(α)
$s \sim D(n) :=$ XOF(ρ)
$e \sim \mathcal{D}_{\sigma}^{m}$
$A \sim R_q^{n\times m}$
for $i = 0$ **to** k **do**
  for $j = 0$ **to** n **do**
    $A[i][j] := (pk[i] ora[j])$
  end for
end for
$p\_t \sim P(m) :=$ G($(A.s + e), s\_t$)
return (($pk := (\sigma, \alpha,$ p_t) , sk:= (s, s_t))

In the provided code, the equation

$$A \cdot s + e = p\_t$$

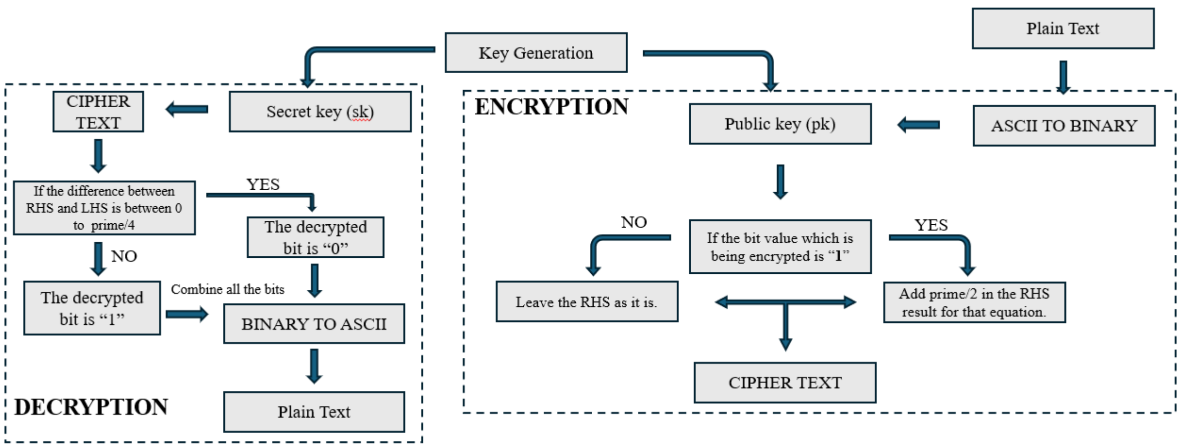

Figure 1 – Schematic of the proposed quantum-resistant encryption for secure end-to-end communication.

represents the encryption process. Where,

- A is a matrix used in the encryption process made from pk, a, where each row of A is generated with the help of boolean OR operation of elements between a and pk.

- s is the secret key vector made up of both sk and sk_t , where sk_t represents the dimensions that are excluded from multiplication with matrix A to calculate the resultant vector present in t. Excluding dimensions in s_t increases the hardness to reduce the lattice for short vectors.

- e is an error vector, which introduces randomness and ensures that the encryption process is secure its range is being calculated by the Gaussian function.

- pk_t is the encrypted text vector which is then compressed and sent as a public key(pk).

3.2 Encryption and Cipher Text

The encryption function is defined to encrypt a given message. It iterates over each character of the message, converts it to binary representation, encrypts each bit using the public key, and generates encrypted text based on the encryption algorithm. Fig. 1 shows the schematic of the proposed quantum-resistant encryption for secure end-to-end communication. The code iterates over each character of a given message (s), converts it to binary representation, encrypts the binary representation using the encryption function, and stores the encrypted text in the vector array. The plain text "m" then first we have to convert into binary so "M" = "1 0 1 1 0 0 1", Where M is an array containing the binary bits for a message m. Now, each of its bits will randomly take any "γ" number of rows of "A" and "pk_t" and add them to form a new set of equations. So, here let's suppose at random we chose any number of rows then for the 1st bit the equation will be. And since this bit is equal to "1" we will also add in the RHS side with the resultant a q/2 extra term and then modulo with q else 0 will be added.

Algorithm 2 Enc(pk, m): Encryption

$M \leftarrow \mathcal{B}_m^{32}$
$dim \leftarrow D(\gamma) := \beta_k^{\gamma}$
$a \leftarrow$ XOF(σ) and $p \leftarrow$ XOF(α)
$u \sim R_q^{M\times n}$ and $v \sim R_q^{M}$
for $l = 0$ **to** size of M **do**
  $u[l] = \sum_{i\in dim} \sum_{j=0}^{256} (a[j]$ or $p[l])$
  $v[l] = \sum_{i\in dim} p\_t[i]_q + M \times [q/2]$
end for
return $c := (u, v)$

3.3 Decryption

The decryption algorithm attempts to recover the original text from the cipher text (c:= (u, v)). Since the cipher text contains two matrices, each row of matrix u contains the coefficients of n dimensions for each bit and matrix v contains the modulo resultant for each of the single bits. So, The decryption algorithm takes each row separately and passes through function H, in which the dot product of a secret vector with each row is done and the redundant dimensions results are removed ($\sum_{j=0}^{n} s.u[i][j] - \sum_{j=0}^{k} s[s\_t].u[i][j]$). Then we consider the difference between the result we got with the corresponding row of the v vector. Then check the probability if the difference is less than $\frac{q}{4}$ to get the bit 0 or 1. Combining all the bits we get our original text message.

Algorithm 3 Dec(sk = s, c = (u , v)): Decryption

$m := B($size of u$)$
for $i = 0$ **to** size of u **do**
  $m[i] = if(Pr[H(v[i] - \sum_{j=0}^{256} s.u[i][j])] < [q/2]) :=$ 0
  else $m[i] = 1$
end for
return m

4. CORRECTNESS AND SECURITY

Correctness: In the absence of errors within the LWE samples provided, the inner product between the error e =

$\langle a, s\rangle$ and the secret s would result in either 0 or $\lfloor \frac{q}{2} \rfloor$, depending on the encrypted bit. Consequently, the decryption process remains consistently accurate. Thus, decryption errors only arise if the summation of error terms across all secrets exceeds $\frac{q}{4}$. Given that we sum at most m normal error terms, each with a standard deviation αq, the standard deviation of the sum remains at most $\sqrt{m}\alpha q < \frac{q}{\log n}$. An analysis confirms that the likelihood of such a normal variable surpassing $\frac{q}{4}$ is negligible.
Now, let's delve into the security proof, demonstrating the system's resilience according to the LWE assumption. Assume vulnerability to chosen plaintext attacks, signifying that for a non-negligible portion of secrets s, there exists an efficient method to accurately predict the encrypted bit, given a public key $(a_i, b_i)_{i=1}^{m}$ selected from the LWE distribution and encryption of a random bit as described earlier, with a probability of at least $\frac{1}{2} + \frac{1}{\text{poly}(n)}$. Given our parameter set and algorithm, many attacks appear futile.

Core-SVP Hardness: The SVP hardness depends upon how much time to solve for the given parameter set more likely the number of equations, modulus size, and the number of dimensions. Various lattice reduction techniques allow to production of the shortest vector in exponential time complexity. No such known algorithm now even with the help of quantum algorithms can solve a particular LWE problem in polynomial time. Blum, Kalai, and Wasserman's work [23] yields a far more intriguing approach that takes just $2^{O(n)}$ samples and time. The method works by breaking down the n coordinates into $\frac{1}{2}$ $log(n)$ blocks of size $2n/log(n)$ each, and then building S recursively by finding collisions in these blocks. This is based on a clever idea that finds a small set S of equations (let's say, of size n) among $2^{O(n)}$ equations, such that $\sum_S a_i$ is, let's say, (1, 0,..., 0). By summing those equations, we get a very noisy estimate for the first coordinate of s, but a typical computation still reveals a bias of around $2^{\theta(n)}$ toward the correct value. Therefore, by repeating the preceding technique just $2^{O(n)}$ times, we have a good likelihood of recovering the value of s1 and thus all of the s.
Blum et al. approach is the most well-known technique for solving the LWE problem. The best-known algorithms for lattice problems [24], [25], take $2^{O(n)}$ time, which is closely connected to this.

Primal attack: The primary attack involves creating a distinct SVP instance from the LWE problem and applying BKZ to solve it. We investigate the minimum block dimension b that BKZ needs to find the unique solution. Given LWE instance (A, b = As+e) then one make a lattice $\Lambda = \{ x \in \mathbb{Z}^{m+1}\ (A|I_m| - b)x = 0 \bmod q \}$ and then it will generate a distinct or unique solution v = (s, e, 1) of norm $\lambda = \zeta\sqrt{kn+m}$ where the ζ shows the standard deviation[25][16][12].

Success condition. With our parameter set at first it will be very difficult to make an accurate Λ or the basis matrix

$$B = \begin{bmatrix} I_{n\times n} & \text{A'} & 0 \\ 0_{m-n\times n} & qI_{(m-n)\times(m-n)} & 0 \\ c^T & & t \end{bmatrix} \in \mathbb{Z}^{(m+1)\times(m+1)}$$

Since we have used redundant dimensions our c matrix which is (A.s + e) doesn't show the exact results for all the dimensions rather it shows the lattice of (n - M) dimensional lattice with error so that the original c matrix is hard to retrieve which will affect the lattice reduction technique to guess the correct set of vector and for even with the efficient algorithm it would take more than O(n^2) to search for each possibility. Even if it guessed the dimensions and got some c the condition of success will be Let e^* be the projection of e orthogonally onto the first $d-b$ vectors of the Gram-Schmidt basis B^*.
BKZ-like algorithms will call an SVP oracle on the last block of dimension b.
If e^*_{d-b} is the shortest vector in that block, it will be found.
If e^*_i is the shortest vector for all projections up to $d-b$, it will "travel to the front".
Assume $\|e^*_{d-b}\| \approx \sigma \cdot \sqrt{b}$.
Applying the GSA, we expect the shortest vector to be found in the last block to have a norm

$$\|b^*_{d-b+1}\| = \alpha_{d-b} \cdot \delta^0_d \cdot \text{Vol}(B)^{\frac{1}{d}} = \delta^0_{-2(d-b)} \cdot \delta^0_d \cdot \text{Vol}(B)^{\frac{1}{d}}$$
$$= \delta^0_{2b-d} \cdot \text{Vol}(B)^{\frac{1}{d}}.$$

Thus, we expect success if

$$\sigma \cdot \sqrt{b} \leq \delta^0_{2b-d} \cdot \text{Vol}(B)^{\frac{1}{d}}.$$

And for the parameters, we numerically optimized it to defend against these types of attacks. Our parameter set allows security against BKZ attacks. The time complexity for these attacks is still exponential but it's one of the best-known attacks to uSVP.

Side-channel attacks: The implementation of our algorithm is such that side-channel attacks don't seem much of a help in tracing the modulo or any timing leakage. We implemented it in such a way that there is a tight bound of timing and is nearly a constant time. Even if the matrix generation from the seed doesn't leave any timing leakage, there would be a case of leakage in modular reduction but it can be easily modified by implementing the algorithm on the device.

Hybrid attacks: A hybrid attack that combines Meet-in-the-Middle combinatorial search with lattice reduction techniques, might compromise several algorithms. The analysis of this approach is very challenging, and new research indicates that it is frequently less competitive than previously believed. As mistakes and secrets are not ternary and scarce in our design, we see that this attack is particularly pertinent in certain situations.

5. PERFORMANCE ANALYSIS

Indeed, the algorithm produces different encryption text vectors for the same input each time it runs due to the use of random variables. This property is crucial for the security of cryptographic systems, as it ensures that even if an attacker observes multiple ciphertexts of the same plaintext, they cannot deduce any information about the original message

Table 1 – Performance comparison of different algorithms

Algorithm	Secret Key Size	Public Key Size	Ciphertex
Our Algorithm	1053 bytes	965 bytes	8799 by
Kyber768	2400 bytes	1184 bytes	1088 by
Kyber1024	3168 bytes	1568 bytes	1568 by
FRODO	11280 bytes	11296 bytes	11288 b
NTRU-scheme	1422 bytes	1140 bytes	1281 by

without knowledge of the secret key. The sizes of the keys also plays a vital role in selecting the parameters. Table I shows the performance comparison of different algorithms for different sizes of the secret key, public key, and ciphertexts. It can be inferred from Table I that the proposed algorithm uses much smaller keys (both public and secret), however, ciphertext size is slightly higher than other algorithms excluding FRODO.
We have taken the strings of various sizes in kilobytes (KB) of different characters in length and ran our algorithm through various processors. We took the average after "20 iterations" of each test case and found this mean data. We have taken 256 randomized integers in our secret key and each time calculated the encryption and decryption timings. The data shown in Table II in seconds and we are expecting much better results after further optimization. These are done on the Ryzen7 4800h processor.

Size of plaintext	Encryption	Decryption
1 KB	0.02 sec	0.003 sec
4 KB	0.408 sec	0.06 sec
15 KB	1.757 sec	0.25 sec
31 KB	3.64 sec	0.518 sec
59 KB	7.44 sec	1.013 sec
118 KB	14.3 sec	1.976 sec

Table 2 – Time taken for encryption and decryption

Fig. 2 illustrates how the encryption and decryption time gaps increase, indicating that the ciphertext computational time is a little bit longer since we are doing O(nγ) for each bit. To simultaneously minimize the size and temporal complexity, we are optimizing this timing and attempting to decrease it to O(log (n)γ). Also, we compress each coefficient of the equations generated by the random seed so that fewer bytes are being used in storage. Since the implementation of the client-server model requires restrictions on the size of messages sent at one time. That's why it's important to compress the coefficients but accordingly, we have to consider the parameters carefully so that security would not be affected.

6. IMPLEMENTATION

This section provides complete structure of our model of end-to-end (E2E) communication channel via client-server model and how we implemented it in the device. Along with the reference implementation of the algorithm and the required encoding of the algorithm in the PKE system for the E2E channel.
As shown in Fig. 3, the implementation was done on a

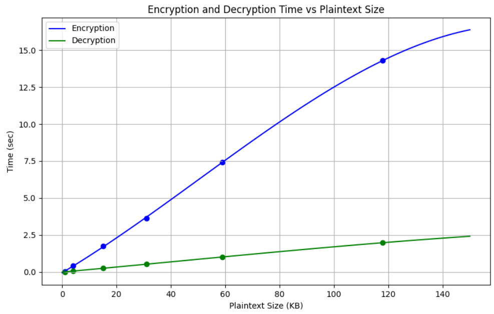

Figure 2 – Time taken (in seconds) for encryption and decryption for different sizes of plaintext (in KB)

client-server model. In which the sender and the receiver have been connected to the devices which are indeed connected with the server that has a database. A new user when comes to the server and registers with the help of the algorithm in the backend will be a unique public key and a private key will be generated on the database for each of the users. The public generated is in the form of of vector of vectors having three vectors of pk_t, pk, a, working as a seed vector that can able to generate the whole matrix A with the help of a and pk.
Now, for the generation of the cipher text from a plaintext that a sender (client 1 in fig.) wants to send to the receiver (client 2 in fig.) the encryption algorithm (Algorithm 2) works in the backend, firstly, it makes an API call to get the public key from the database. Using the public key pk of the receiver the senders create the cipher text Enc($\mathbf{pk}$) to get the cipher text and send it back to the server and store it in the database as in encrypted format. When the server realises there is a change it makes a call back to the receiver's end and the API traces the change and reflects it to the receiver's side. Now, since we got the cipher text and now with the help of Algorithm 3 (Decryption) the previously stored secret key at the receiver's end comes to use to decrypt this cipher text using Dec($\mathbf{c}, \mathbf{sk}, \mathbf{sk\_t}$)
The use of the local storage here is to keep the secret keys in it with the help of cookies, we parse the sk and sk_t with cookie tokens to the local storage. After that, we could access that token form the backend and retrieve the secret keys out of that which will able to decrypt the incoming messages and show us in the chat box. Now, since everything is running smoothly there is still an issue of storage of ciphertext since the text size is about $\mathbf{0(nm)}$ where the m denotes the number of bits per character generally '7'. So, it's approx 1800 integers each taking approx 8-12 bytes making an overall space of 7.2 KB and for 250 such characters which is the current character limit for our algorithm implementation in the chat server, it comes to approx 1.8 MB which turns out to be a lot as of now. So, to reduce this we are implementing a $nlogn$ space size algorithm which would encrypt multibit at the same time. To reduce space it would also make the runtime lesser.
Further work on user and software interfaces and key management systems may be necessary for encryption using QKD devices to guarantee long-term access to encrypted messages and also asynchronicity. The demand is presently not being met by the key rates, particularly when it comes

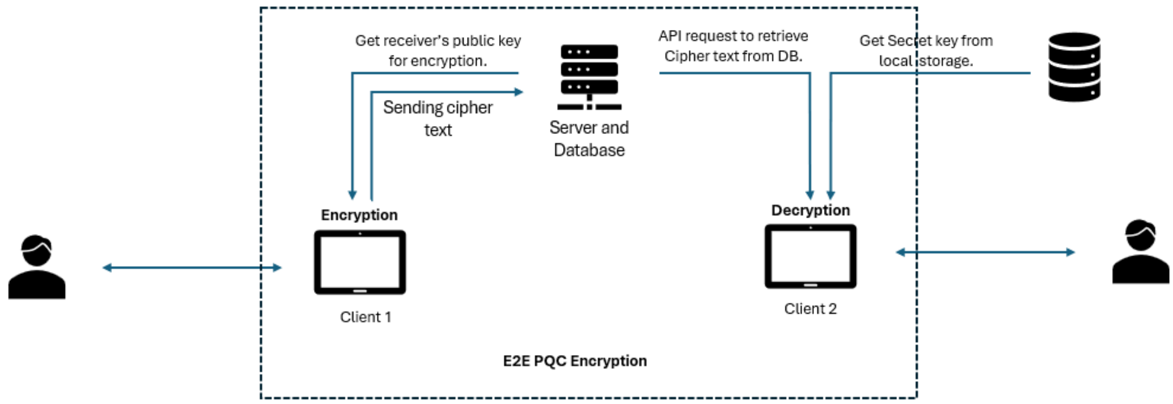

Figure 3 – Schematic of the proposed quantum-resistant encryption for secure end-to-end communication.

to the integration of the Quantum key distributions into systems other than the site-to-site VPNs. This issue may be resolved by deriving many keys from the original key that QKD supplied. For instance, one may utilize a single QKD key with keys that are generated for every message transmitted between two fixed participants on the same day. It could also go beyond the classical client-server model and has many use cases such as a secure quantum channel for the military or some esteemed organization. Since we are successfully able to integrate it into the system and establish secure communication between two parties. There can be still some key exchanges and session chatting features which we are introducing further.

6.1 Computational Complexity for Attacker

The difficulty of breaking an encryption scheme based on Learning With Errors (LWE) is closely tied to solving the Shortest Vector Problem (SVP) for the lattice defined by the encryption parameters. As lattice dimensionality increases, the computational complexity of solving SVP grows exponentially, making it extremely challenging for attackers. Randomly choosing dimensions for each bit and adding random error terms during encryption further increase complexity. The hardness of breaking an LWE-based encryption scheme depends on lattice dimensionality, the modulus used, and the number of random dimensions chosen during encryption. Solving SVP in high-dimensional lattices is believed to be exponentially complex, even for powerful computers. In a client-server model, LWE-based encryption secures encrypted chats stored in the database, making decryption without keys extremely difficult due to the complexity of solving SVP and the randomness introduced. Brute-forcing through the more than 10^{1200} possible combinations in LWE encryption makes chosen ciphertext attacks (CCA) infeasible. Regev's reduction shows that a quantum solver for the LWE problem with certain parameters can solve approximate versions of lattice problems like SVP and SIVP over lattices of dimension n. However, this reduction doesn't imply that these lattice problems are NP-Hard for linear approximation factors. Even if a quantum solver for LWE worked for parameters very close to 1, the resulting approximation factor for lattice problems would still be greater than n. Currently, we lack proof that these approximate lattice problems are NP-Hard for linear approximation factors. Therefore, while a quantum polynomial-time algorithm for LWE would place LWE within BQP, it wouldn't directly impact classical hardness or NP-hardness questions. Such an algorithm would still be a significant breakthrough. The approximate versions of lattice problems aren't known to be NP-Hard, and efficient algorithms for solving them remain elusive.

7. CONCLUSION AND FUTURE WORK

The security of this lattice-based encryption scheme relies on the difficulty of solving the Shortest Vector Problem (SVP) in high-dimensional lattices, particularly when random dimensions and error terms are introduced. The algorithm generates a random basis, random dimensions, and random errors for each bit, making it computationally infeasible for an attacker to recover the secret key or decrypt the ciphertext efficiently. However, the size and storage of ciphertexts and messages impose restrictions on the length of messages that can be sent at once. To address this, we are exploring various transformations and researching methods to reduce the ciphertext size to a logarithmic scale, possibly through binary search algorithms. Additionally, we are investigating techniques similar to the Number Theoretic Transform (NTT) for optimizing matrix multiplications, which could enhance performance without compromising security.

REFERENCES

[1] Oded Regev, "The Learning with Errors Problem" in *Blavatnik School of Computer Science, Tel Aviv University*.

[2] Katharina Boudgoust, Corentin Jeudy, Adeline Roux-Langlois, Weiqiang Wen "On the Hardness of Module Learning with Errors with Short Distributions," Journal of Cryptology, 2023, .

[3] David Balbás, "The Hardness of LWE and Ring-LWE: A Survey," *Universidad Politécnica de Madrid*, 8th October 2021.

[4] C. Peikert., ". Public-key cryptosystems from the worstcase shortest vector problem.," *In Proc. 41st ACM Symp. on Theory of Computing (STOC), pages 333– 342. 2009*.

[5] Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, Damien Stehlé, "Algorithm Specifications And Supporting Documentation (version 3.01)," in *NIST PQC round 3*, January 31, 2021 .

[6] Daniele Micciancio Oded Regev, "Lattice-based Cryptography, IEEE, 2008.

[7] Huck Bennett†, "The Complexity of the Shortest Vector Problem," in *Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 170 (2022)*.

[8] Ajtai and Dwork, "A public-key cryptosystem with worst-case/average-case equivalence" in *STOC '97: Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*.

[9] Joppe Bos, Léo Ducas†, Eike Kiltz‡, Tancrède Lepoint§, Vadim Lyubashevsky¶, John M. Schanck, Peter Schwabe, Gregor Seiler††, Damien Stehlé‡‡, , "CRYSTALS – Kyber: a CCA-secure module-lattice-based KEM" in *2018 IEEE European Symposium on Security and Privacy*.

[10] Applebaum, B.; Cash, D.; Peikert, C.; Sahai , "A. Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems" in *Proceedings of the Advances in Cryptology-Crypto, International Cryptology Conference, Santa Barbara, CA, USA, 16–20 August 2009*.

[11] Lei Bi, Xianhui Lu, Junjie Luo, Kunpeng Wang Zhenfei Zhang , "Hybrid dual attack on LWE with arbitrary secrets" https://doi.org/10.1186/s42400-022-00115-y.

[12] Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, Damien Stehlé , "ACRYSTALS-Kyber Algorithm Specifications And Supporting Documentation (version 3.01)" in *NIST round 3 submission https://pq-crystals.org/kyber/data/kyber-specification-round3-20210131.pdf*.

[13] Christoph Döberl, Wolfgang Eibner ,Simon Gärtner, Manuela Kos, Florian Kutschera Sebastian Ramacher "Quantum-resistant End-to-End Secure Messaging and Email Communication" in *ARES '23: Proceedings of the 18th International Conference on Availability, Reliability and Security*.

[14] Uddipana Dowerah, Srinivasan Krishnaswamy "Towards an efficient LWE-based fully homomorphic encryption scheme", https://doi.org/10.1049/ise2.12052.

[15] UAndrea Basso "Lattice-based cryptography and SABER", in *Quantum CS seminar, Budapest, 25 March 2021*.

[16] Yu Wei1,2, Lei Bi1,2* , Xianhui Lu1,2 and Kunpeng Wang1,2 "LSecurity estimation of LWE via BKW algorithms", in https://cybersecurity.springeropen.com/articles/10.1186/s42400-023-00158-9.

[17] Miklós Ajtai, Ravi Kumar, D. Sivakumar "A sieve algorithm for the shortest lattice vector problem", in *STOC '01: Proceedings of the thirty-third annual ACM symposium on Theory of computing July 2001*.

[18] Andreas Hülsing, Joost Rijneveld, John Schanck, and Peter Schwabe. "High-speed key encapsulation from NTRU". In Wieland Fischer and Naofumi Homma, editors, in *Cryptographic Hardware and Embedded Systems – CHES 2017, LNCS. Springer, 2017*.

[19] Ziyu Zhao, Jintai Ding. "Practical Improvements on BKZ Algorithm", in NIST.

[20] Jeffrey Hoffstein, Jill Pipher, Joseph H. Silverman. "NTRU: A Ring-Based Public Key Cryptosystem ", https://www.ntru.org/f/hps98.pdf.

[21] Shi Bai, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler and Damien Stehlé "CRYSTALS-Dilithium Algorithm Specifications and Supporting Documentation", https://pq-crystals.org/dilithium/data/dilithium-specification-round3-20210208.pdf.

[22] Shay Gueron1,2 and Fabian Schlieker3

"Speeding up R-LWE post-quantum key exchange ", *https://eprint.iacr.org/2016/467.pdf*.

[23] A. Blum, A. Kalai, and H. Wasserman, "Noise-tolerant learning, the parity problem, and the statistical query model ", in *Journal of the ACM, 50(4):506–519, 2003.*.

[24] M. Ajtai, R. Kumar, and D. Sivakumar "A sieve algorithm for the shortest lattice vector problem. ", in *In Proc. 33rd Annual ACM Symp. on Theory of Computing (STOC), pages 601–610. 2001*.

[25] D. Micciancio and P. Voulgaris. "A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. ", in *In STOC. 2010*.

[26] Xue Zhang, Zhongxiang Zheng Xiaoyun Wang "A detailed analysis of primal attack and its variants. ", https://link.springer.com/article/10.1007/s11432-020-2958-9