Post-Quantum Module Learning with Rounding-Based Public Key Encryption Using Incomplete Number Theoretic Transform

Anupama Arjun Pandit[1], Arun Mishra[2]*
Department of Computer Science and Engineering,
Defence Institute of Advanced Technology, Girinagar, Pune, 411025, Maharashtra, India[1,2]
Department of Computer Engineering, Marathwada Mitra Mandal's College of Engineering,
Pune, 411052, Maharashtra, India[1]

***Abstract*—Post Quantum Cryptographic (PQC) techniques are widely used in encryption standards and digital signatures. Lattice-based post-quantum cryptographic techniques have been reported in the last decades. The present work proposes an optimized quantum-safe lattice public key encryption (PKE) scheme based on the Module Learning with Rounding (MLWR) problem, enhanced by the use of Incomplete Number Theoretic Transform (NTT). The objective of the proposed scheme is to achieve efficient encryption and decryption while maintaining robust security in accordance with the National Institute of Standards and Technology (NIST) recommendations. The incomplete NTT relaxes the modulus q requirement, enables a smaller modulus for efficient arithmetic, and reduces computational complexity. This approach results in significant improvements in the speed of key generation, encryption, and decryption with a marked reduction in rejection probability, compared to schemes utilizing complete NTT. The proposed scheme demonstrates competitive performance against other lattice-based encryption schemes such as Kyber and Frodo. It shows lower encryption and decryption times while offering comparable security levels. Proposed scheme is at least a hundred times faster than Frodo lattice-based public-key encryption schemes. For NIST-recommended security level, in proposed scheme, each encryption needs an average of 300K CPU cycles, and each decryption needs 120K CPU cycles. Additionally, modulus 7937 enables a reduction in key and ciphertext sizes, optimizing the scheme for practical deployment in resource-constrained environments. Performance evaluations confirm the practicality of the scheme with substantial reductions in computational overhead, making it a highly efficient and secure candidate for post-quantum encryption.**

***Keywords*—Post-quantum cryptography; public key encryption; lattice-based cryptography; Incomplete Number Theoretic Transform; Module Learning with Rounding; Learning with Errors**

I. Introduction

The rise of quantum computing poses a critical challenge to traditional cryptographic systems, which largely depends on the difficulty of solving mathematical problems like prime factorization and discrete logarithms. These problems form the basis of many widely used encryption schemes, but quantum algorithms, particularly Shor's algorithm [1], can solve them in polynomial time. This has led to an urgent need for cryptographic techniques that can resist quantum attacks, thus fueling the development of post-quantum cryptography (PQC). Since the National Institute of Standards and Technology (NIST) launched an initiative to develop new encryption standards that are unbreakable by quantum computers, PQC [2], [3] is getting more and more attention. One of the most appealing areas in PQC is lattice-based cryptography (LBC). LBC grabbed the theoretical community's interest because LBC structures are accompanied by security proofs ingrained in worst-case instances. Ajtai and Dwork [4] proposed 1^{st} lattice-based encryption algorithm. Regev later simplified and improved on this approach in [5]. The formulation of an intermediary problem, Learning with Errors (LWE) was one of Regev's key contributions. LWE was asymptotically at least as hard as certain traditional worst-case lattice problems [6], [5], and reasonably straightforward to utilize in cryptographic constructs.

LWE involves sampling errors using Gaussian distributions to introduce noise, which forms the backbone of its cryptographic constructs. Banerjee et al. [7] created learning with rounding (LWR) problem, which is an LWE variant with derandomization; LWR creates an instance utilizing deterministic rounding under a small modulus instead of introducing random errors to make it noisy. LWR instance sampling is simpler than LWE instance sampling since it does not use the Gaussian sampling technique. Several recent studies on the hardness of LWR problem have concluded that LWR problem is at least as difficult as the LWE problem when the number of samples is limited [8].

Schemes based on LWE variant require sampling from error distributions for randomness. Also, errors introduced into ciphertexts and public keys inevitably result in increased bandwidth. Since the error is made predictably, LWR-based schemes reduce the bandwidth by using a small rounding modulus. The module variants of the problems enable interpolation of the original LWR/LWE problems and their ring variants, reducing bandwidth and complexity of computation. It is possible to increase the level of security of Ring-LWE/LWR without making any changes to the basic mathematics, as demonstrated by the modules [6], [9].

One key observation in LBC is the dominance of polynomial multiplication as the most time consuming operation in these schemes. LBC relies on high-degree polynomials, making polynomial multiplication the bottleneck for efficiency. Fig. 1 highlights the time profiles of various LBC schemes, confirming that polynomial multiplication significantly impacts performance.

*Corresponding author.

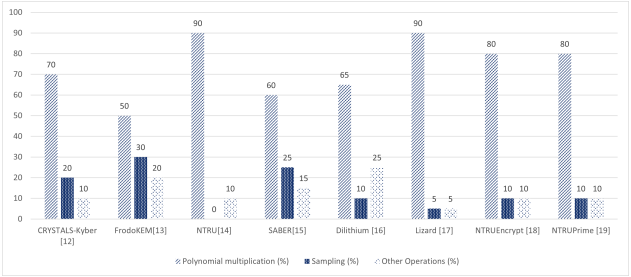

Fig. 1. Time-consuming operations in lattice-based cryptographic schemes [10], [11], [12], [13], [14], [15], [16], [17].

Efficient implementation of polynomial multiplication methods can significantly reduce the computational overhead in LBC schemes, making these schemes more practical for real-world applications.

Lattice-based schemes, particularly those employing the module variants of LWR, strike an effective balance between bandwidth reduction and computational complexity. They facilitate efficient polynomial operations through techniques like Number Theoretic Transform (NTT), a method for fast polynomial multiplication in finite fields. However, the traditional NTT approach can become computationally expensive when polynomials are large. To address this, Incomplete NTT [18] introduces a hybrid methodology where NTT process is truncated early for smaller sub-polynomials allowing alternative, simpler multiplication methods like the schoolbook algorithm to handle smaller parts. This truncation reduces computational overhead and makes the scheme more efficient without compromising security.

Current proposal: Present study proposes an efficient encryption scheme which uses an optimization technique that combines Module Learning With Rounding (MLWR) with Incomplete NTT for polynomial multiplication. The proposed PKE scheme comprises of generation of keys, encryption, and decryption algorithms. The time needed to create keys and complete the decryption procedure is incredibly quick when compared to other lattice-based PKEs. Joel Alwen et.al.'s LWR properties are taken into consideration while choosing parameters for key generation, encryption, and decryption [9]. In the proposed scheme, a public matrix is created using an expansion function that uses the generalized hash function Shake-256 [19]. Additionally, the proposed scheme incorporates the Montgomery Reduction Algorithm [20], which optimizes modular arithmetic by transforming the modulus into a more computationally manageable form. This transformation enables faster computations during key generation, encryption, and decryption processes by reducing the cost of modular multiplications. The Incomplete NTT and Montgomery Reduction Algorithm are employed in the proposed approach to accelerate matrix multiplication and to reduce modulus, respectively. Using the MLWR offers an optimal ground between security and efficiency [21] in comparison to the original LWR and Ring-Learning with Rounding (RLWR) problems. The proposed scheme is more efficient than earlier methods because it also decreases the size of a public key by using a rounding function. For a recommended security level, proposed scheme's public key size is 1632 bytes. Choice of the underlying hard problem, as well as the scheme's parameters (Secret distribution range η, modulus q, polynomial ring $(X^{256}+1)$, and rank of module k), are inspired by 3 key principles: simplicity of the scheme, efficiency, and flexibility.

Technical novelty of proposed work lies in the specific hybrid integration of MLWR problem with an Incomplete NTT using the prime modulus $q = 7937$. While existing lattice-based schemes utilize either standard LWE with NTT (e.g., Kyber) or MLWR with alternative multiplication methods (e.g., SABER), our construction leverages a unique synergy. By setting the truncation level to $\beta = 1$, we halt the butterfly operations early, which reduces computational cycles while maintaining robust security margins. This specific optimization avoids the final, most computationally expensive stage of decomposition, offering a tailored balance between performance and quantum resistance.

Section II gives the necessary background information; Section III presents related work. Section IV describes proposed PKE with correctness and security; Section V discusses implementation details; Section VI discusses the result and its comparison with other lattice-based PKEs; and in Section VII we state the conclusion and future work.

II. Preliminaries

A. Notations

$\mathbb{Z}_q$ is the integer ring under modulo q, and $\mathbb{Z}$ mod q is $\mathbb{Z}$'s reduction in $[0, q)$. With $N = 256$, R_q is quotient ring $\mathbb{Z}_q[x]/(x^N + 1)$. R_q^k is the k-dimensional quotient ring under modulus q. Uppercase bold alphabets are used for matrices, such as "matrix $\boldsymbol{A}$", whereas lowercase regular alphabets are used for scalars, such as "scalar p", and lowercase bold alphabets are used for vectors, such as "vector $\boldsymbol{s}$". The symbol $\mathcal{L}(B)$ denotes the lattice produced by basis set B. Sample vector $\boldsymbol{a}$ from the distribution $\mathcal{D}$ is denoted by $\boldsymbol{a} \leftarrow \mathcal{D}$. The sample vector $\boldsymbol{a}$ according to a centered binomial distribution β_i is denoted by $\boldsymbol{a} \leftarrow \beta_i$. Primitive N^{th} root of unity denoted by ζ. Primitive $2N^{th}$ root of unity denoted by Ψ.

Convolution: For polynomials **a,b** in $\mathbb{Z}[x]$ with length N. Convolution simplifies to

$$c_k = \sum_{i=0}^{k} \mathbf{a}_i \mathbf{b}_{k-i}, \quad k = 0, \ldots, 2N-2$$

This corresponds to the coefficient-wise multiplication of the polynomials.

- Cyclic Convolution: equivalent to polynomials and multiplication in $\mathbb{Z}[x]/(x^N - 1)$
- NWC (Negative Wrapped Convolution): is used for polynomial multiplication when polynomials are in the quotient ring $\mathbb{Z}[x]/(x^N + 1)$

In the context of NTT, the arrangement of input and output coefficients plays a critical role in the efficiency of algorithms. Two common types of orderings are:

- Normal Order (no): Coefficients are arranged in increasing natural order (e.g., $[x_0, x_1, x_2, \ldots, x_{n-1}]$)
- Bit-Reversed Order (bo): Coefficients are rearranged such that their indices correspond to the bit-reversed representation of the original indices

We adopt the notation from [22] for the Cooley-Tukey NTT algorithm. Specifically:

$$NTT^{CT,\Psi}_{no\rightarrow bo,\beta=1}(\mathbf{x})$$

where,

- NTT^{CT}: Indicates the Cooley-Tukey algorithm is used.
- Ψ: Refers to the twiddle factors ($2N-th$ roots of unity) used in NTT the computation.
- $no \rightarrow bo$: Specifies that the input coefficients are in natural order (no), while output coefficients are in bit-reversed order (bo).
- $\beta = 1$: Implies a number of levels cropped in NTT computation.

B. Lattices

A lattice is an infinitely expanded structure of discrete points arranged in a repeating pattern. The lattice is constructed by an integer linear combination of independent basis vectors. Set B is a collection of N-linearly independent vectors. The dimension of the lattice is N. let $B = b_1, b_2, b_3, b_4, b_5, \cdots, b_N$

$$\mathcal{L}(b_1, b_2, b_3, b_4, b_5, \cdots, b_N) = \left\{ \sum_{j=1}^{N} y_j b_j : y_j \in \mathbb{Z} \ \& \ b_j \in \mathbb{R} \right\}$$

C. Module Lattices

Module lattices extend the concept of lattices to modules over a ring. Let R be a ring, and k be a rank of R-module. A module lattice $\mathcal{L}$ in R^k is a discrete set of points in R^k defined as Eq. (1):

$$\mathcal{L} = \left\{ \sum_{i=1}^{k} z_i \mathbf{b}_i : z_i \in R \right\}, \tag{1}$$

where, $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k \in R^k$ are the basis vectors of the module lattice.

D. Learning with Errors (LWE)

LWE [5] for N-dimensions, α error rate, and q as a modulus denoted as $\text{LWE}_{N,q,\alpha}$ and defined as follows:
From given samples $(\boldsymbol{a}_j, b_j) \in \mathbb{Z}_q^N \times \mathbb{Z}_q$, where: $\boldsymbol{a}_j$ is chosen uniformly at random, and b_j is computed as:

$$b_j = \langle \boldsymbol{a}_j, \boldsymbol{s} \rangle + e_j \mod q$$

with $\langle \boldsymbol{a}_j, \boldsymbol{s} \rangle$ representing the inner product of $\boldsymbol{a}_j$ and a secret vector $\boldsymbol{s} \in \mathbb{Z}_q^N$ and e_j is a small error term sampled from a error distribution χ. The LWE problem is to recover the secret vector $\boldsymbol{s}$ by solving a given noisy linear equation.

The LWE problem can be expressed in matrix form as:

$$\boldsymbol{A}\boldsymbol{s} + \boldsymbol{e} = \boldsymbol{b} \mod q,$$

where, $\boldsymbol{A} \in \mathbb{Z}_q^{N \times N}$ is a matrix with rows $\boldsymbol{a}_j$

E. Learning with Rounding (LWR)

Banerjee et al. [7] first presented the LWR problem for enhancing efficiency of the LWE-based pseudorandom generator (PRG). It was used to build reusable computational extractors, lossy trapdoor functions, and deterministic encryption. In fact, the LWR problem may be considered as a deterministic alternative to LWE problem. The only difference is that the error in LWR is deterministic, which makes it less random than the Gaussian error in LWE. Error in the LWR is caused by a scaling operation in the rounding function. In particular, the error in LWE is β-bounded ($\beta > 2\sqrt{N}$) for security [8], while error in LWR is less than $1/2$. Scaling $\mathbb{Z}_q$ down to $\mathbb{Z}_p$ generates the deterministic error, where p is a rounding modulus.

The scaled rounding function [12] is defined like this:

$$\lfloor \cdot \rceil_p \text{ is } \mathbb{Z}_q \rightarrow \mathbb{Z}_p : p < q \ .$$

For matrix $\boldsymbol{A}$, and vector $\boldsymbol{s}$ a sample of LWR is obtained by:

$$(\boldsymbol{A}, b = \lfloor < \boldsymbol{A}, \boldsymbol{s} > \rceil_p)$$

F. Module Learning with Rounding (MLWR)

MLWR is a module variant of LWR, specifically tailored for cryptographic applications. It combines the modular arithmetic of lattices with the deterministic rounding technique of LWR to achieve efficient and secure constructions.

In the MLWR problem, the objective is to compute the approximate inner product of a secret vector $\mathbf{s}$ with a public matrix $\mathbf{A}$ while reducing noise through deterministic rounding. Formally, for a modulus q and a smaller modulus p, the MLWR sample is generated as:

$$\mathbf{b} = \left\lfloor \frac{p}{q} \cdot (\mathbf{A} \cdot \mathbf{s}) \right\rceil \mod p$$

where,

- $\mathbf{A} \in \mathbb{Z}_q^{k \times k}$ is a public matrix.
- $\mathbf{s} \in \mathbb{Z}_q^k$ is a secret vector.
- $\lfloor \cdot \rceil_p$ denotes deterministic rounding to the nearest multiple of p.

G. Number Theoretic Transform (NTT)

NTT [23] is a discrete Fourier transform (DFT) defined over finite fields. It is a crucial tool for efficient polynomial multiplication in cryptographic schemes, especially in LBC. The NTT maps polynomial coefficients into a point-value representation using N-th roots of unity modulo a prime q.

NTT enables fast polynomial multiplication with integer coefficients and is widely employed in LBC schemes. For the NTT to be well-defined, the modulus q must satisfy $q \equiv 1 \pmod{2N}$, ensuring the existence of a primitive N-th root of unity ζ modulo q.

The goal of NTT is to multiply two polynomials having integer coefficients such that the computed polynomial coefficients lie under a specific module. The advantage of NTT is that it provides a fast technique for multiplication whereas the disadvantage is that NTT is possible with prime modulo of the form like $2^a \cdot b + 1$ where a and b are arbitrary constants.

Given a polynomial $a(x) = \sum_{i=0}^{N-1} a_i x^i \in Z_q[x]/\langle x^N - 1 \rangle$, the NTT transforms it into its evaluation at N-th roots of unity modulo q.

For NTT to be well-defined and invertible, the following conditions must hold:

- The modulus q must be a prime number such that $q \equiv 1 \pmod{2N}$
- A primitive N-th root of unity ζ must exist modulo q, i.e. $\zeta^N \equiv 1 \mod q$

Forward NTT transforms a polynomial $a(x)$ from its coefficient representation into its evaluation (point-value) representation. Forward NTT of $a(x)$ is computed as Eq. (2):

$$\hat{a}_k = \sum_{j=0}^{N-1} a_j \zeta^{jk} \pmod{q}, \quad k = 0, 1, \ldots, N-1 \tag{2}$$

The inverse NTT (iNTT) transforms the point-value representation back to coefficient form [see Eq. (3)]:

$$a_j = \frac{1}{N} \sum_{k=0}^{N-1} \hat{a}_k \zeta^{-jk} \pmod{q}, \quad j = 0, 1, \ldots, N-1 \tag{3}$$

Here, N^{-1} is a modular multiplicative inverse of N modulo q.

Efficient computation of the NTT relies on algorithms that exploit symmetries in the evaluation process. The Fast Fourier Transform (FFT) [24] is an efficient algorithm to compute DFT of a sequence. In context of polynomial multiplication, FFT is instrumental in reducing the computational complexity from $O(N^2)$ to $O(N \log N)$. The FFT decomposes a polynomial into simpler components using a divide-and-conquer strategy, exploiting the properties of roots of unity.

The FFT process involves recursively splitting the polynomial ring $Z_q[x]/(x^N + 1)$ into smaller subrings. At each stage, the polynomial is divided into two halves using specific roots of unity. For example, the ring $Z_q[x]/(x^N + 1)$ is split into $Z_q[x]/(x^{N/2} - \Psi^{N/2})$ and $Z_q[x]/(x^{N/2} + \Psi^{N/2})$, where Ψ is primitive $2N^{th}$ root of unity. This process is repeated until the polynomial is reduced to base components, such as $Z_q[x]/(x - \Psi)$ or $Z_q[x]/(x + \Psi)$, which are directly computable.

Two widely used approaches of FFT are the Cooley–Tukey(CT) [25] and Gentleman–Sande(GS) [26] algorithms:

- Cooley–Tukey(CT) Algorithm: The CT algorithm is also known as the Decimation-In-Time (DIT). It performs the transformation by dividing input sequence into even and odd indices. Key step involved in the butterfly operation is:

$$a' = a + \zeta b, \quad b' = a - \zeta b,$$

where, ζ is the twiddle factor (root of unity).

The butterfly operation for Cooley–Tukey is illustrated in Fig. 2. This method starts with the input in natural order and produces an output in bit-reversed order.

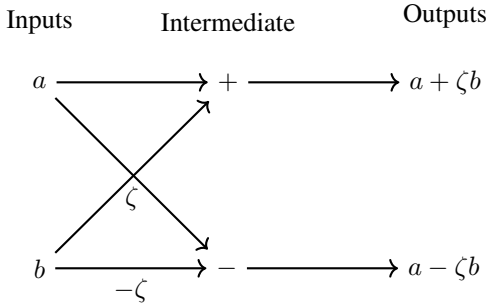

Fig. 2. Cooley-Tukey butterfly.

- Gentleman–Sande (GS) Algorithm: The GS algorithm is also known as the Decimation-In-Frequency (DIF). It follows a similar divide-and-conquer approach but starts with the coefficients of the polynomial arranged in a bit-reversed order. Inverse NTT is often implemented using this algorithm, as it efficiently reconstructs the original sequence from its Fourier-transformed counterpart. The butterfly operation in Gentleman-Sande is given by:

$$a' = a + b, \quad b' = (a - b) \cdot \zeta^{-1}$$

Fig. 3 illustrates the Gentleman–Sande butterfly operation. This algorithm efficiently reconstructs the original sequence from its transformed version.

Both algorithms use butterfly operations to combine intermediate results efficiently.

NTT Applications in Polynomial Multiplication: Polynomial multiplication in cryptographic schemes often involves convolutions. Using NTT, the multiplication of two polynomials $a(x)$ and $b(x)$ modulo $x^N - 1$ is performed as:

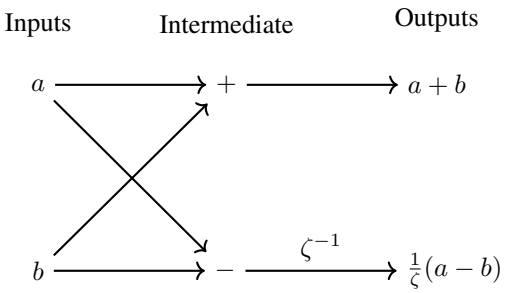

Fig. 3. Gentleman-Sande butterfly.

- Compute the NTT of $a(x)$ and $b(x)$: $\hat{a} = NTT^{CT,\zeta}(a(x))$, $\hat{b} = NTT^{CT,\zeta}(b(x))$ refer Eq. (2).
- Perform pointwise multiplication: $\hat{a}_k \circ \hat{b}_k \pmod q = \hat{c}_k, \ \forall k = 0, 1, \ldots, N-1$, $\circ$ denotes pointwise multiplication in NTT domain.
- Compute inverse NTT of $\hat{c}$ to obtain product polynomial in coefficient form: $c(x) = iNTT^{GS,\zeta^{-1}}(\hat{c})$ refer Eq. (3).

This process reduces the complexity of polynomial multiplication from $O(N^2)$ to $O(N \log N)$, making it highly efficient for cryptographic applications.

NTT has been integrated into the specifications of several post-quantum cryptographic schemes, including Kyber, Saber, and Dilithium. These schemes leverage their efficiency to perform fast polynomial arithmetic, enabling secure and efficient Public Key Encryption (PKE) and signature generation.

H. Incomplete NTT

Incomplete NTT method is employed for polynomial multiplication when dimension (N) of the lattice is a power of two and modulus q is an NTT-friendly prime but does not satisfy $q \equiv 1(mod 2N)$. An incomplete NTT method does not have to map $Z_q[x]/(x^N + 1)$ down to linear terms, and one can stop its mapping before the last β level, $\beta = 0, 1, \cdots, \log N - 1$

Mathematical Formulation: Given two polynomials $\mathbf{x}$ and $\mathbf{y}$ in $Z_q[x]/(x^N + 1)$, their product $\mathbf{r}$ using incomplete NTT is computed as:

$$\mathbf{r} = \text{INTT}_{bo\to no}^{GS,\Psi^{-1}} \left(\text{NTT}_{no\to bo,\beta=1}^{CT,\Psi}(\mathbf{x}) \circ \text{NTT}_{no\to bo,\beta=1}^{CT,\Psi}(\mathbf{y}) \right),$$

where:

- $\text{NTT}_{no\to bo,\beta=1}^{CT,\Psi}$: Forward Cooley-Tukey NTT [25] up to $\beta = 1$ level
- $\text{INTT}_{bo\to no}^{GS,\Psi^{-1}}$: Inverse Gentleman-Sande NTT [26] to reconstruct the coefficients

Fig. 4 illustrates the polynomial splitting process in the incomplete NTT, showing how the polynomial ring $Z_q[x]/(x^N + 1)$ is gradually decomposed into smaller components till it reduces to degree two polynomials. The decomposition stops early, reducing computational overhead while maintaining accuracy. It requires $(\log(N) - 1)$ steps.

Advantages of Incomplete NTT:

- Relaxed Conditions for Modulus q: Incomplete NTT requires only a $\frac{N}{2}$-th root of unity for cyclic NTTs or an N-th root of unity for negacyclic NTTs, significantly increasing the range of valid q.
- Computational Efficiency: As it stops decomposition at an earlier level, incomplete NTT reduces the number of butterfly operations, saving additions and subtractions at the cost of slightly more complex base multiplications.
- Adaptability: The degree of incompleteness β can be tailored to optimize performance for specific platforms or resource constraints.

The forward incomplete NTT algorithm transforms a polynomial into its NTT domain representation using CT butterfly operations. Unlike complete NTT, which fully decomposes the polynomial down to linear terms, incomplete NTT stops earlier, saving computations and providing more freedom in parameter selection. The algorithm operates as follows:

- Initialization: Start with the full polynomial length N and segment it into smaller parts.
- Butterfly Operations: Apply Cooley-Tukey butterfly operations to each segment. For each pair of coefficients, compute their sum and difference, scaled by a twiddle factor, which is a precomputed power of a $2N$-th root of unity Ψ.
- Modulo Reduction: Ensure all intermediate results remain within the finite field $\mathbb{Z}_q$ by applying modulo q.

The inverse incomplete NTT reconstructs the coefficients from their partially transformed representation. The steps include:

- Initialization: Begin with segments of size 1 and progressively combine them to reconstruct the full polynomial.
- Butterfly Operations: Combine pairs of coefficients, using inverse twiddle factors and ensuring results remain within $\mathbb{Z}_q$.
- Normalization: After reconstruction, normalize all coefficients by multiplying with N_inv, the modular multiplicative inverse of N.

The incomplete NTT provides a flexible and computationally efficient alternative to complete NTT for polynomial arithmetic in cryptography. By halting decomposition at a higher level β, it reduces computational costs while relaxing constraints on the modulus q. This hybrid approach is particularly beneficial for resource-constrained environments like embedded systems and is implemented in schemes such as Kyber to achieve optimal performance.

III. Related Work

Lattice-based cryptography has emerged as a promising candidate for efficient and secure PQC schemes. The security of LBC is based on worst-case hard problems such as the Shortest Vector Problem (SVP) [27], Closest Vector Problem (CVP) [28], approximate Shortest Independent Vectors Problem (SIVP) [6], LWE [5] and Small Integer Solution (SIS) [6]. These problems provide robust security guarantees due to their inherent computational difficulty.

Early lattice-based PKE schemes, such as the Ajtai-Dwork cryptosystem [29], relied on SVP for security. However these schemes required high-dimensional lattices, leading to impractically large key sizes. The introduction of the NTRU cryptosystem [12] marked a significant milestone, offering compact key sizes and security rooted in polynomial rings. Later, Regev's LWE-based scheme [5] provided further improvements in efficiency, though it exhibited limited resistance to active attacks like chosen plaintext attack(CPA). Subsequent works, including Lindner and Peikert's PKE [30], reduced key and ciphertext sizes by leveraging optimized parameters.

In 2016, NIST initiated a call for PQC algorithm standardization [2], propelling the development of LBC. Notable candidates such as Kyber, Saber, and Frodo emerged, each demonstrating unique strengths:

- Kyber [10], a finalist in NIST PQC project, is based on Module-LWE(MLWE) problem. It incorporates NTT for efficient polynomial multiplication, enabling fast and memory-efficient implementations. Kyber's use of centered binomial distributions for sampling and its compatibility with NTT-friendly parameters make it highly optimized for modern cryptographic applications.

- Saber [13] relies on the MLWR problem, which eliminates error sampling and reduces randomness requirements compared to MLWE-based schemes. Saber uses alternative polynomial multiplication methods such as Karatsuba and Toom-Cook due to its power-of-two moduli, which are incompatible with traditional NTTs.

- Frodo [11] employs standard LWE and matrix-based multiplication over $\mathbb{Z}_q$. Although Frodo offers strong theoretical security guarantees, it suffers from higher bandwidth and computational requirements compared to structured lattice-based schemes.

MLWR problem offers multiple benefits that make it a compelling choice for cryptographic schemes:

- Simplified Sampling: Unlike MLWE, MLWR eliminates the need for error sampling, reducing computational overhead and simplifying implementations.

- Optimized Bandwidth: By using a small rounding modulus, MLWR achieves smaller ciphertext and key sizes, making it suitable for bandwidth-constrained environments.

- Security Guarantees: MLWR maintains security equivalence to MLWE under specific parameter constraints, offering robust protection against quantum attacks.

- Implementation Flexibility: MLWR-based schemes can be efficiently implemented using alternative polynomial multiplication techniques like NTT and Karatsuba, providing flexibility in resource-constrained environments.

The aforementioned benefits make MLWR a highly attractive foundation for designing practical and secure LBC systems, particularly in modern applications requiring both efficiency and scalability.

Proposed approach bridges the gap between the efficiency of power of two rounding in SABER and the fast NTT arithmetic found in Kyber. The novelty over SABER is a use of Incomplete NTT for faster polynomial multiplication, whereas the novelty over Kyber is the selection of $q = 7937$. Mathematically, $q = 7937$ is highly advantageous; it satisfies $q \equiv 1 \pmod{256}$, ensuring NTT compatibility, yet yields a rejection probability of only 0.031. This is a significant improvement over standard NTT-friendly primes

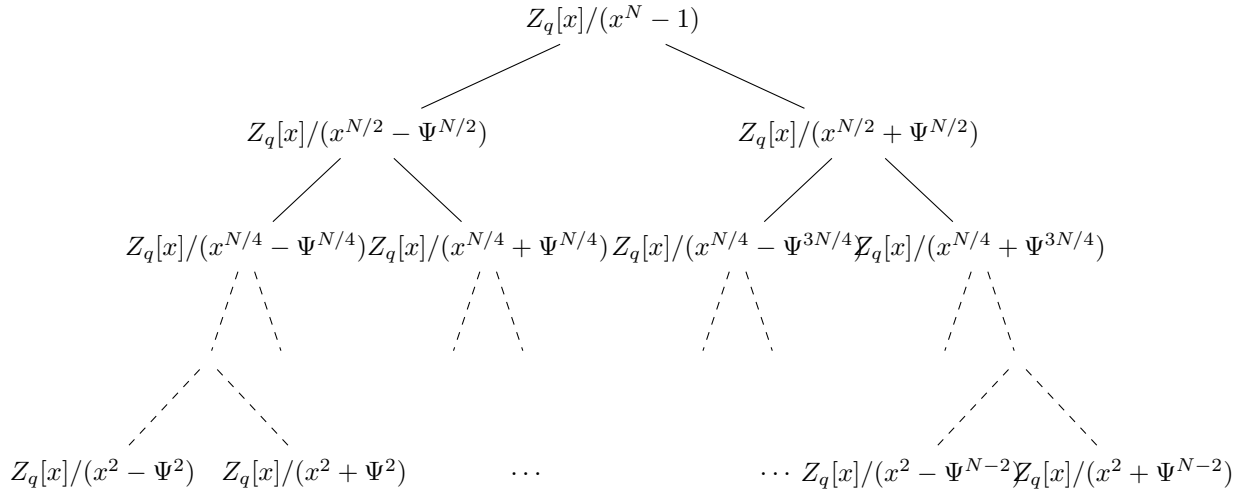

Fig. 4. Polynomial splitting in incomplete NTT.

like $q = 3329$ (rejection probability 0.594), directly optimizing the key expansion and sampling processes. Consequently, the proposed scheme achieves competitive execution times while minimizing the randomness requirements inherent in MLWE-based constructions.

A. NTT in Lattice-Based Cryptography

Efficient polynomial multiplication is a critical component of LBC. NTT and its inverse (INTT) play a pivotal role in modern implementations. We analysed NTT-based implementations in prominent NIST PQC schemes such as Kyber [10], Saber [13], NTRU [12], Dilithium [14], and Falcon [31]. These schemes adopt diverse strategies to optimize polynomial multiplication based on their underlying ring structures and security requirements.

NTT is particularly efficient for schemes with moduli satisfying $q \equiv 1 \pmod{2N}$, as seen in Kyber, Falcon, and Dilithium. Conversely, schemes like Saber and NTRU employ alternative techniques due to their use of power-of-two moduli or non-NTT-friendly rings. For example, Kyber transitioned from full NTT in its initial rounds to an incomplete NTT approach in later rounds, leveraging optimized Fast Fourier Transform (FFT) techniques to reduce computational complexity. NTT-unfriendly rings refer to polynomial rings where the structure of the ring does not align well with the standard requirements of NTT, such as specific polynomial degrees or forms. Common examples include rings where the degree N of the polynomial is not a power of two.

Incomplete NTT is a specialized variant of the traditional NTT that halts the decomposition process at an intermediate stage, skipping the final levels of polynomial transformation. This method reduces the number of butterfly operations required in Cooley-Tukey or Gentleman-Sande algorithms, significantly lowering computational complexity. Additionally, the incomplete NTT relaxes modulus requirements, enabling the use of smaller moduli, 7937. The reduced modulus allows for more efficient arithmetic operations, optimized memory usage, and smaller key and ciphertext sizes. These advantages make the incomplete NTT particularly best-suited for resource-constrained environments, such as IoT devices [32], where computational efficiency and reduced bandwidth are paramount.

B. Proposed Approach: Combining MLWR and Incomplete NTT

The proposed PKE scheme integrates the advantages of MLWR and the computational efficiency of incomplete NTT. MLWR reduces bandwidth and randomness requirements by eliminating error sampling, as demonstrated in Saber. Meanwhile, incomplete NTT minimizes computational overhead while maintaining compatibility with lattice-based polynomial arithmetic. These optimizations align with insights from Falcon [31] and Kyber [10], where NTT-based methods significantly enhance performance. By adopting this hybrid approach, the proposed scheme offers a post-quantum cryptographic solution with balanced security and efficiency.

IV. Proposed MLWR-Based PKE using Incomplete NTT

From the analysis of existing NTT in LBC, we have chosen the most efficient Incomplete FFT-based NTT for implementation of MLWR-based PKE. The first advantage of the proposed scheme is that it can reduce the ciphertext size by mapping coefficients from a larger ring $\mathbb{Z}_q$ to a smaller ring $\mathbb{Z}_p$. The second advantage is encryption and decryption times and the required CPU cycles are comparable to other prominent lattice-based candidates of the NIST PQC project.

Proposed PKE scheme incorporates the Incomplete NTT for efficient polynomial operations in LBC. The process begins with generating cryptographic parameters which are used to create a public key (for encryption) and a secret key (for decryption). A plaintext message is encrypted with a public key, producing a ciphertext. The decryption process uses a secret key $\boldsymbol{s}$ to recover original message. Incomplete NTT is applied during both encryption and decryption to optimize computational performance. This design ensures secure communication while enhancing efficiency in lattice-based cryptosystems.

Proposed scheme is made up of the following three algorithms:

A. MLWR Key Generation Algorithm

The proposed key generation algorithm constructs a public-private key pair for the MLWR-based PKE scheme. The process is detailed as follows:

- Seed Generation: A cryptographic seed denoted as $seed\_\delta$, is generated using the SHAKE-256 hash function. This seed is a part of the public key pk and is used to generate a $k \times k$ public matrix $\boldsymbol{A}$.
- Matrix Construction: The matrix $\boldsymbol{A}$ is constructed with elements as polynomials in a ring $R_q = \mathbb{Z}_q[x]/(x^N + 1)$.

  Here:

  - q is a prime modulus suitable for incomplete-NTT operations.
  - $N = 256$, the degree of the polynomial, satisfies the incomplete-NTT and LWR specifications.
- Secret Key Generation: The secret key $\boldsymbol{s}$ is an N-dimensional short vector sampled from the ring R_q. Coefficients of $\boldsymbol{s}$ are chosen from a small range to ensure efficiency and maintain security.
- Public Key Computation: Public key attribute $\boldsymbol{b}^T$ is computed as:

$$\boldsymbol{b}^T = \left\lfloor \frac{p}{q} \cdot (\boldsymbol{A} \cdot \boldsymbol{s}) \right\rceil_p$$

  where: The multiplication $\boldsymbol{A} \cdot \boldsymbol{s}$ is optimized using the *Incomplete NTT* method, and p is the rounding modulus, smaller than q.

- Output: Public key pk consists of seed $seed\_\delta$ and k-dimensional MLWR sample $\boldsymbol{b}^T$. The private key sk is the short vector $\boldsymbol{s}$.

The pseudocode for key generation algorithm is presented in Algorithm 1.

Algorithm 1 KeyGeneration($seed\_\delta$)

1: $seed\_\delta \leftarrow D\{0,1\}^N$ ▷ Generate a random binary seed.
2: $\boldsymbol{s} \leftarrow Ds_\eta^k$ ▷ Sample secret key $\boldsymbol{s}$ from distribution Ds_η^k
3: $\boldsymbol{A} \in R_q^{k\times k} := Expansion(seed\_\delta)$ ▷ Expand the matrix $\boldsymbol{A}$ from the seed.
4: $\boldsymbol{As} = \text{INTT}_{bo\to no}^{GS,\Psi^{-1}}\left(\text{NTT}_{no\to bo,\beta=1}^{CT,\Psi}(\mathbf{A}) \circ \text{NTT}_{no\to bo,\beta=1}^{CT,\Psi}(\mathbf{s})\right)$ ▷ Efficiently compute $\boldsymbol{As}$ using incomplete NTT.
5: $\boldsymbol{b^T} = \lfloor \frac{p}{q}\boldsymbol{As}\rceil_p \in R_p^{k\times 1}$ ▷ Apply rounding operation
6: return $(pk := (seed\_\delta, \boldsymbol{b^T}), sk := (\boldsymbol{s}))$ ▷ Output public and private key pair.

B. MLWR Encryption Algorithm

The encryption algorithm in the proposed MLWR-based PKE scheme efficiently transforms the plaintext into a ciphertext while ensuring performance. The steps are as follows:

- Plaintext Representation: The plaintext is represented as a $mbit$.
- Matrix Construction: Seed $seed\_\delta$ is used to construct the matrix $\boldsymbol{A}$, whose elements belong to the ring $R_q = \mathbb{Z}_q[x]/(x^N+1)$
- Random Vector Sampling: A random N-dimensional vector $\boldsymbol{a}$ is sampled. This vector is then multiplied with the matrix $\boldsymbol{A}$ using the *Incomplete NTT* method, to optimize the computational efficiency. The result is stored in $\boldsymbol{u}$:

$$\boldsymbol{u} = \boldsymbol{A}\cdot\boldsymbol{a}$$

- Ciphertext Calculation: Using the public key attribute $\boldsymbol{b^T}$ and the plaintext message bit $mbit$ ($messagespace \in 0,1$), the ciphertext $\boldsymbol{c}$ is computed as:

$$\boldsymbol{c} = \boldsymbol{b^T a} + mbit\cdot\left\lfloor\frac{q}{2}\right\rfloor$$

  $mbit\cdot\lfloor\frac{q}{2}\rfloor$ operation encodes original message.
- Output: The encryption function outputs ciphertext $\boldsymbol{c}$ and the preamble $\boldsymbol{u}$.

Pseudocode for the encryption is presented in Algorithm 2.

C. MLWR Decryption Algorithm

The decryption process in the proposed MLWR-based PKE scheme is the final step, responsible for recovering the original plaintext message. The procedure involves the following steps:

- Intermediate Value Calculation: Using the secret key $\boldsymbol{s}$, the algorithm computes the intermediate value d_1 as:

Algorithm 2 MLWREncryption(pk,mbit)

1: $\boldsymbol{A} \in R_q^{k\times k} := Expansion(seed\_\delta)$ ▷ Expand the matrix $\boldsymbol{A}$ from the seed.
2: $\boldsymbol{a} \leftarrow Ds_\eta^k$ ▷ Sample vector $\boldsymbol{a}$ from distribution Ds_η^k
3: $u = \text{INTT}_{bo\to no}^{GS,\Psi^{-1}}\left(\text{NTT}_{no\to bo,\beta=1}^{CT,\Psi}(\mathbf{A}) \circ \text{NTT}_{no\to bo,\beta=1}^{CT,\Psi}(\mathbf{a})\right)$ ▷ Compute $\boldsymbol{Aa}$ using incomplete NTT.
4: $c = [\boldsymbol{b^T a} + mbit\cdot\lfloor\frac{q}{2}\rfloor]$ ▷ Add scaled message to the rounded public key component.
5: return (u, c) ▷ Output the ciphertext components u and c.

$$d_1 = \boldsymbol{u}\cdot\boldsymbol{s}$$

Here, $\boldsymbol{u}$ is the preamble received with the ciphertext.

- Decrypted Value: The decrypted value d is obtained by subtracting the computed d_1 from the ciphertext $\boldsymbol{c}$:

$$d = \boldsymbol{c} - d_1$$

All operations are performed modulo q.
- Plaintext Recovery: The range of $d \mod q$ is checked to determine the original plaintext bit ($mbit$):
  - If $\frac{-q}{4} \le d \le \frac{q}{4}$, the plaintext $mbit$ is set to $\mathbf{0}$
  - If $\frac{q}{4} < d \le \frac{3q}{4}$, the plaintext $mbit$ is set to $\mathbf{1}$

Algorithm 3 presents the pseudocode for decryption.

Algorithm 3 MLWRDecryption($\boldsymbol{s}, c, u$)

1: $d_1 = \text{INTT}_{bo\to no}^{GS,\Psi^{-1}}\left(\text{NTT}_{no\to bo,\beta=1}^{CT,\Psi}(\mathbf{u}) \circ \text{NTT}_{no\to bo,\beta=1}^{CT,\Psi}(\mathbf{s})\right)$ ▷ Compute $\boldsymbol{us}$ using incomplete NTT.
2: $d = c - d_1$ ▷ Subtract the private component from the ciphertext.
3: **if** $(((d \ mod \ q) \ge \frac{-q}{4})$ *AND* $((d \ mod \ q) \le \frac{q}{4}))$
4: **then** ▷ Check if d lies in the range
5: $d = 0$
6: **else**
7: **if** $(((d \ mod \ q) > \frac{q}{4})$ *AND* $((d \ mod \ q) \le \frac{3q}{4}))$
8: **then**
9: $d = 1$
10: return d ▷ Output the decrypted message.

D. Correctness of the Scheme

The correctness of the decryption algorithm relies on ensuring that the introduced rounding error remains within acceptable bounds.

If (u, c) is a correctly formulated ciphertext, the decrypted value is given as Eq. (4):

$$\begin{aligned} d &= c - d_1 \qquad \text{from Algorithm 3} \\ &= c - \boldsymbol{s}^T u \\ &= \boldsymbol{b^T a} + mbit\cdot\frac{q}{2} - \boldsymbol{s}^T\boldsymbol{Aa} \qquad \text{from Algorithm 2} \end{aligned} \tag{4}$$

Here, $\boldsymbol{b^T}$ is defined during key generation process as:

$$\boldsymbol{b^T} = \left\lfloor \frac{p}{q} \boldsymbol{s}^T \boldsymbol{A} \right\rceil \mod p$$

This rounding operation can be rewritten as an LWE problem:

$$\boldsymbol{b^T} \approx \boldsymbol{s}^T \boldsymbol{A} + \frac{p}{q} e_r \pmod q,$$

where, $\frac{p}{q} e_r$ introduces a random error into $\boldsymbol{s}^T \boldsymbol{A}$, and $p < q$. Let $\frac{p}{q} e_r = e_{rnd}$, then:

$$\boldsymbol{b^T} \approx \boldsymbol{s}^T \boldsymbol{A} + e_{rnd}$$

By substituting $\boldsymbol{b^T} = \boldsymbol{s}^T \boldsymbol{A} + e_{rnd}$ into Eq. (4):

$$\begin{aligned} d &= (\boldsymbol{s}^T \boldsymbol{A} + e_{rnd})\boldsymbol{a} + mbit \cdot \lfloor \frac{q}{2} \rfloor - \boldsymbol{s}^T \boldsymbol{A}\boldsymbol{a} \\ &= \boldsymbol{s}^T \boldsymbol{A}\boldsymbol{a} + e_{rnd}\boldsymbol{a} + mbit \cdot \lfloor \frac{q}{2} \rfloor - \boldsymbol{s}^T \boldsymbol{A}\boldsymbol{a} \\ &= e_{rnd}\boldsymbol{a} + mbit \cdot \lfloor \frac{q}{2} \rfloor \end{aligned} \tag{5}$$

To analyze correctness, consider two cases based on the value of $mbit$:

a) Case 1: When $mbit = 0$, Eq. (5) simplifies to:

$$d = e_{rnd}\boldsymbol{a}$$

For correctness, the term $||e_{rnd}\boldsymbol{a}||$ must satisfy:

$$||e_{rnd}\boldsymbol{a}|| \in \left[\frac{-q}{4}, \frac{q}{4}\right]$$

Since $||e_{rnd}|| \in \left[\frac{-q}{4}, \frac{q}{4}\right]$ with high probability, correctness is achieved as $||e_{rnd}\boldsymbol{a}||$ lies within the valid range.

b) Case 2: When $mbit = 1$, Eq. (5) becomes:

$$d = e_{rnd}\boldsymbol{a} + \lfloor \frac{q}{2} \rfloor$$

Correctness requires that:

$$||e_{rnd}\boldsymbol{a} + \lfloor \frac{q}{2} \rfloor|| \in \left[\frac{q}{4}, \frac{3q}{4}\right]$$

For this condition to hold, the rounding error term $e_{rnd}\boldsymbol{a}$ must be small enough to offset the added $\lfloor \frac{q}{2} \rfloor$. Hence, the threshold for $||e_{rnd}||$ becomes:

$$||e_{rnd}|| \in \beta = \left[\frac{-q}{4}, \frac{q}{4}\right]$$

By setting $mbit = 0$ or $mbit = 1$, the correctness of the decryption is ensured when the rounding error e_{rnd} remains within the defined threshold. The analysis confirms that $||e_{rnd}\boldsymbol{a}||$ lies within the valid range $\left[\frac{q}{4}, \frac{3q}{4}\right]$ with high probability, ensuring accurate decryption.

Conclusion: As $||e_{rnd}\boldsymbol{a}||$ lies within the valid interval $\left[\frac{-q}{4}, \frac{q}{4}\right]$, the scheme ensures correctness. This analysis highlights the role of rounding errors and the chosen $mbit$ in maintaining the accuracy of decryption.

E. Parameter Selection

Parameter selection is an important part of the proposed scheme. Table I shows selected parameters for proposed PKE:

TABLE I. PARAMETERS FOR THE PROPOSED SCHEME

Parameters	**Value**
Modulus	$q = 7937$
Secret coefficient range η	[-2,2] for high level security, [-3,3] for recommended security
Degree of polynomial ring (N)	256
Polynomial ring	$X^{256} + 1$
Rank of module k	2 (for security level 2), 3 (for security level 3), 4 (for security level 5)

Selection of the modulus q for PKE involves several critical considerations to ensure security, efficiency, and compatibility with lattice-based cryptographic operations. Here's a detailed process to select appropriate parameters. Mathematical Properties:

- Prime Modulus: Typically, q is chosen to be a prime number.
- NTT Compatibility: For efficient NTT-based polynomial multiplication, modulus q should satisfy $q \equiv 1 \mod 256$, where 256 is a degree of the polynomial (typically a power of 2). This ensures existence of primitive 256-th roots of unity modulo q.

Coefficients of the polynomial $\boldsymbol{s}$ and the public matrix $\boldsymbol{A}$ are sampled uniformly from $\mathbb{Z}_q$ using the rejection sampling method [33]. This approach generates random numbers of the required bit length from the SHAKE-128 pseudorandom number generator (PRNG) as candidate values. Only numbers that are strictly less than q are considered valid and accepted. The rejection probability, which quantifies the likelihood of a candidate being discarded, is given by:

$$1 - \frac{q}{2^{\log q}}$$

We have implemented Python code to find out different values of modulus q that satisfy the required mathematical properties. These values are listed in Table II.

Modulus 7937 stands out as an excellent choice for q due to its low rejection probability. The secret coefficient range η determines the distribution from which the coefficients of the secret key are drawn.
In LBC, the security of the system is closely tied to the noise distribution, as it controls the hardness of lattice problems MLWR.

TABLE II. PRIME MODULUS FOR INCOMPLETE NTT

Modulus q	q inverse	256-th root of unity	≈ Rejection probability
3329	-3327	17	0.594
7681	-7679	198	0.062
7937	-7935	71	0.031
9473	-9471	88	0.422
10753	-10751	29	0.344
11777	-11775	50	0.283
12289	-12287	09	0.250

Rejection Probability: It is calculated as $(1 - \frac{q}{2^{\log q}})$. Lower rejection probabilities are preferred for efficiency.

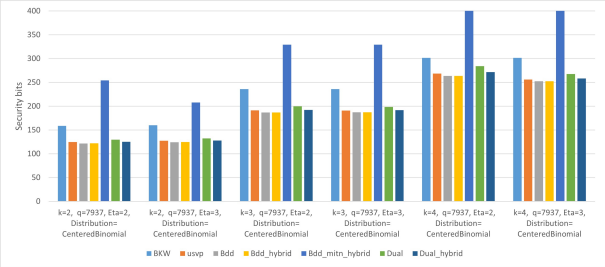

Fig. 5. Choice of secret distribution range.

To estimate the security of different parameter sets of the proposed scheme, lattice estimator [34] is used. The estimator takes parameters as input and gives an estimated security against lattice attacks. Fig. 5 compares the security estimates of various parameter sets under different attack models. These estimates reflect the difficulty of breaking the cryptographic scheme with the given parameters and secret coefficient distribution.

It has been observed that as k increases, the security levels increase across all attack models.

Increasing η slightly decreases security under certain attack models because larger secret coefficients may expose more information.

For lower computational cost and acceptable security, use of $(k = 2, \eta = 3)$. $(k = 3, \eta = 2)$ is a good intermediate choice, balancing security and performance. For maximum security, use $(k = 4, \eta = 2)$ with centered binomial distribution. This offers the highest resistance.

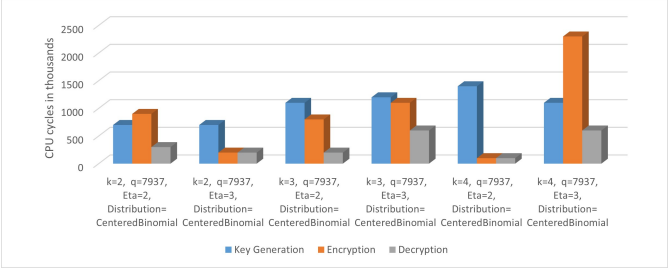

Fig. 6. CPU cycles w.r.t. different distributions.

The given Fig. 6 outlines the computational cost (in CPU cycles) for operations (Key Generation, Encryption, Decryption, and NTT) across different parameter sets.

It shows that higher lattice dimensions $(N \times k)$ lead to increased computational costs for all operations. $N \times k = 512, \eta = 3$ offers the lowest computational cost, particularly for Encryption and Decryption, making it ideal for resource-constrained devices. $N \times k = 768, \eta = 2$ provides a good balance between security and computational cost. $N \times k = 1024, \eta = 2$ is the best choice for applications requiring maximum security, albeit at the cost of higher computation, particularly for NTT.

The selected parameters, which include modulus q=7937, secret coefficient range $\eta \in [2, 3]$, polynomial ring $[x]/(x^{256} + 1)$, module rank k, and incomplete NTT, provide a balanced combination of security, efficiency, and computational performance. The careful selection of these parameters ensures that the proposed MLWR-based PKE scheme meets the requirements for post-quantum cryptographic applications while optimizing execution time and maintaining a high level of security.

- Parameter Selection and Failure Probability: The parameters in Table I are selected to minimize the Decryption Failure Probability (DFP) while maximizing security.
  - Modulus $q = 7937$: This prime satisfies $q \equiv 1 \pmod{256}$, supporting NTT, and provides a low rejection probability of 0.031 [33].
  - Rounding and Noise: We use a centered binomial distribution β_η for secrets. Rounding modulus is chosen such that rounding error e_{rnd} satisfies $||e_{rnd}|| \in [-\frac{q}{4}, \frac{q}{4}]$.
  - DFP Analysis: Decryption is successful if the accumulated noise from rounding does not cross the decision threshold. For the recommended set $(k = 3, \eta = 2)$, the probability of a decryption failure is estimated to be $< 2^{-128}$ [34].

F. Security Estimation and Comparison

This section presents the security estimation of proposed MLWR-based PKE over different moduli. Blum–Kalai–Wasserman (BKW) [34] attack is used to estimate the computing costs of solving corresponding hard problems for security evaluation.

The BKW algorithm, originally developed for solving Learning Parity with Noise (LPN) problem, was extended to the LWE problem. It uses a blockwise approach to reduce the dimension of the problem while managing the induced noise. The process involves sample reduction using sort-and-match techniques, hypothesis testing to distinguish between valid and invalid guesses of sub-vectors, and back substitution to iteratively reduce the problem's dimensionality. The advancements in BKW, such as lazy modulus switching and quantization, have significantly improved its efficiency, making it a central tool in security evaluations of LBC schemes.

Martin Albrecht's lattice estimator tool [34] is used for this task. Table III summarizes the security estimations for proposed PKE scheme against the BKW attack. It includes different parameter sets with specific lattice dimensions 512, different moduli, and an estimated ≈ Cost.

TABLE III. ESTIMATED SECURITY OF DIFFERENT MODULI FOR INCOMPLETE NTT.

Modulus	Required CPU timing	Estimated ≈ Cost
3329	KeyGen: 69, Enc: 172, Dec: 69, NTT: 45483	$2^{190.3}$
7681	KeyGen: 103, Enc: 69, Dec: 69, NTT: 47793	$2^{182.9}$
7937	KeyGen: 103, Enc: 69, Dec: 34, NTT: 4675	$2^{183.4}$
9473	KeyGen: 138, Enc: 103, Dec: 34, NTT: 46103	$2^{176.2}$
10753	KeyGen: 138, Enc: 103, Dec: 34, NTT: 47690	$2^{176.2}$
11777	KeyGen: 310, Enc: 69, Dec: 34, NTT: 54552	$2^{177.5}$
12289	KeyGen: 241, Enc: 69, Dec: 34, NTT: 49138	$2^{178.3}$

Table III provides a comparison of estimated security of different moduli and their performance metrics for an incomplete NTT in an MLWR-based PKE scheme. The key parameters compared are:

Modulus: The modulus used in the polynomial ring.

Required CPU Timing: This includes the CPU timing for key generation (KeyGen), encryption (Enc), decryption (Dec), and NTT operations.

Estimated Approximate Cost: The security level against the BKW attack is represented as a cost in terms of computational effort (expressed as powers of 2).

The modulus 7937 offers the best decryption time (34), which is crucial for practical applications. It also maintains a competitive NTT timing. 7937 still provides a robust security level at $2^{183.4}$, which is sufficient for most practical purposes. Therefore, the modulus 7937 strikes the best balance between performance and security, making it an optimal choice for the MLWR-based PKE scheme.

1) Formal security analysis: Hardness of the Module Learning with Rounding (MLWR) problem serves as the foundation for the security of the proposed PKE. Security in Indistinguishability under Chosen Plaintext Attack(IND-CPA) is avaluated.

Theorem 1. If the $MLWR_{k,N,q,p}$ problem is hard, then the proposed PKE is IND-CPA secure.

Proof:

Proof proceeds via a sequence of games:

- Game 0: This is the original IND-CPA security game. The public key is $(seed_\delta, \mathbf{b}^T)$ where $\mathbf{b}^T = \lfloor \frac{p}{q}\mathbf{s}^T\mathbf{A} \rceil$ [21].
- Game 1: The MLWR sample $\mathbf{b}^T$ is replaced with a vector $\mathbf{u}$ sampled uniformly at random from R_p^k. Under the MLWR assumption, no polynomial-time adversary can distinguish Game 1 from Game 0 with non-negligible advantage [21], [35].
- Game 2: The ciphertext components (u, c) are replaced with uniform samples from R_p^k and R_p. Since the public key is now uniform, the message $mbit$ is perfectly masked.

The advantage of an adversary $\mathcal{A}$ is bounded by $Adv_{PKE}^{ind-cpa}(\mathcal{A}) \leq Adv_{MLWR}^{dist}(\mathcal{B})$, where $\mathcal{B}$ is a simulator solving the MLWR decision problem.

Reduction confirms that the only way for an adversary to break IND-CPA security of the proposed PKE is by solving a underlying MLWR decision problem. With parameter set $(k = 3, \eta = 2)$, the computational cost to perform such an attack exceeds $2^{183.4}$ core-SVP operations, thereby ensuring robust quantum-resistant security.

V. IMPLEMENTATION DETAILS OF MLWR-BASED PKE USING INCOMPLETE NTT

The first step in the proposed approach is to generate a matrix $\boldsymbol{A} \in R_q^{k\times k}$ using a random seed $seed\_\delta\{0,1\}^{256}$. The hash function SHAKE-128 [19] is used in the current study to expand the seed to matrix $\boldsymbol{A}$, producing enough unpredictability via a high probability. In the encryption process, Matrix $\boldsymbol{A}$ is helpful for both key creation and multiplication operations. Incomplete NTT was utilized to speed up multiplication. The condition of the scheme is that $\boldsymbol{A}$ must be evenly sampled. The expansion function outputs a uniform matrix $\boldsymbol{A}$ in NTT format with coefficients ranging from 0 to $q-1$.

The result of the expansion function is the matrix $\boldsymbol{A} \in R_q^{k\times k}$ from a 256-byte seed input. It converts a uniform 256-byte seed to a uniform matrix $\boldsymbol{A}$ value using the NTT representation. Matrix $\boldsymbol{A} \in R_q^{k\times k}$ is uniformly random with $k = 4$ and elements are polynomials under a ring $\mathbb{Z}_q[x]/(x^N + 1)$. N =256 is the polynomial ring's degree and k is the rank of a module that is used to find out the dimension of the fundamental lattice problem $N \cdot k$.

Rejection sampling on the integer stream generated by the SHAKE-128 is part of the matrix $\boldsymbol{A}$ generation process. Once the matrix $\boldsymbol{A}$ is generated, the next step involves sampling a short vector $\boldsymbol{s}$ through rejection sampling [33]. This sampled vector $\boldsymbol{s}$ will be used as a secret key.

The public key pk contains 2 parts, 1^{st} part is seed $seed\_\delta$ and the second part is $\boldsymbol{b^T} = \lfloor \frac{p}{q}\boldsymbol{s}^T\boldsymbol{A} \rceil_p$. Matrix multiplication is the most important process in the scheme. Present work uses the NTT representation of $\boldsymbol{A}$ and $\boldsymbol{s}$ to carry out pointwise multiplication. Then perform inverse NTT transform on the above multiplication result and also multiply with Montgomery factor for modular reduction. Modular reduction uses the Montgomery algorithm. Rounding $\boldsymbol{b^T}$ operation reduces the size of a public key pk. Minimize size of pk from R_q to R_p in LWR, where modulus $q > p$. The modular reductions necessary for ring R_q arithmetic in the proposed method never use the % (mod) operator. Instead, use special modulus q-specific reduction techniques and Montgomery reductions without the corrective phases. The rounding functions have been calculated using branching-free methods. To make sure that the findings are accurately represented in the reduced form, the Montgomery factor $2^{32} \ mod \ q$ is included in the previously computed roots. Montgomery reductions are used in the NTT domain representations after the pointwise product of the polynomials.

The implementation of the incomplete NTT involves various trade-offs between code size and speed, particularly significant for different hardware environments. On embedded processors, the focus is on minimizing storage and memory requirements, such as avoiding additional temporary storage during polynomial multiplications in R_q. To achieve this, precomputed tables of powers of a primitive root (ψ) are used

to streamline operations, while forward NTT (from normal to bit-reversed order) and inverse NTT (from bit-reversed to normal order) are implemented efficiently.

A. Implementation Security and Side-Channel Mitigation

While primary focus of this work is performance optimization, implementation incorporates several design principles to mitigate side-channel vulnerabilities :

- Constant-Time NTT: The Incomplete NTT is implemented with a fixed execution path [36]. There are no data-dependent branches or secret-dependent loops, which prevents leakage of timing information during polynomial multiplication.
- Montgomery Reduction: We utilize a constant-time Montgomery reduction [20]. By incorporating the Montgomery factor $2^{32} \pmod q$ into the precomputed NTT twiddle factors [26], we avoid the conditional subtraction step typically found in modular reduction, thereby mitigating simple power analysis and timing attacks.

VI. Result and Analysis

A. Experimental Setup

Proposed scheme and comparative benchmarks were implemented in C language, compiled with the GCC compiler. To ensure a fair and standardized evaluation of performance, all experiments were conducted under identical environmental conditions on a CDAC-PARAM Shavak machine. The hardware features an Intel(R) Xeon(R) Gold 6226R v4 series processor. The testing was performed on a Centos operating system, providing a stable platform for measuring precise computational overhead, including execution times and CPU cycle consumption for the proposed and reference LBC schemes.

B. Performance with Different Parameter Sets

Advantage of MLWR is a same polynomial ring R_q is used for all security levels [2]. To attain a higher level of security, the scheme employs a module with increased rank. Table IV presents proposed scheme's performance and security w.r.t different parameter sets. Parameter sets, denoted as P_1, P_2, and P_3, correspond to increasing security levels.

Results highlight the balance between security and performance. Higher security levels lead to increased computational overhead, reflected in greater CPU time and cycle consumption. The selection of an appropriate parameter set is crucial for achieving the desired security level.

TABLE IV. Evaluation of Proposed Scheme with Various Parameter Sets.

Parameter set and Security level	Operation	Average CPU time (μs)	Average CPU cycles
P_1 Medium : ($q = 7937, N = 256, \eta = 3, k = 2, l = 2$)	Key Generation	103	700
	Encryption	65	200
	Decryption	34	100
P_2 Recommended: ($q = 7937, N = 256, \eta = 2, k = 3, l = 3$)	Key Generation	207	1200
	Encryption	68	200
	Decryption	34	100
P_3 High : ($q = 7937, N = 256, \eta = 2, k = 4, l = 4$)	Key Generation	172	1400
	Encryption	69.48	300
	Decryption	34.12	120

C. Security Estimation and Attack Analysis

Table V provides a comprehensive security profile for the proposed MLWR-based PKE scheme across three distinct parameter sets (P_1, P_2, P_3). This analysis moves beyond heuristic estimates to include rigorous metrics against standard lattice attacks [34].

- Configuration Parameters (k and η):
  - Rank of a module k defines lattice dimension as $N \times k$, where $N = 256$ is fixed.
  - Increasing k from 2 to 4 significantly enhances the security margin against all attack vectors.
  - The secret distribution range η is utilized for sampling the short vector $\mathbf{s}$, directly influencing the hardness of the underlying MLWR problem.
- Attack Metrics:
  - Core-SVP: Represents lower bound of complexity for a Shortest Vector Problem in a sieving regime, which is the foundational hard problem for proposed scheme.
  - Primal (uSVP) Attack: Evaluates the complexity of recovering the secret key by treating the MLWR instance as a unique shortest vector problem in an expanded lattice.
  - Dual Attack: Estimates the computational effort required for an adversary to distinguish the MLWR samples from a uniform distribution by finding short vectors in the dual lattice.
  - BKW Attack: Originally for the Learning Parity with Noise problem, it uses a blockwise approach to reduce problem dimensions while managing induced noise through sort-and-match techniques.
- NIST Security Level Mapping:
  - Parameter set P_1 achieves $2^{118.4}$ security bits, corresponding to NIST Category I.
  - The recommended set P_2 ($k = 3, \eta = 2$) provides $2^{183.4}$ bits of security, satisfying Category III.
  - The high-security set P_3 achieves $2^{253.9}$ bits, aligning with Category V for maximum quantum resistance.

The results demonstrate that by utilizing modulus $q =$

TABLE V. SECURITY ESTIMATES AND NIST MAPPING FOR PROPOSED MLWR PARAMETERS

Parameter Set	k	η	Core-SVP	Primal (uSVP)	Dual Attack	BKW	NIST Category
P_1	2	3	$2^{118.4}$	$2^{134.5}$	$2^{139.9}$	$2^{183.4}$	I
P_2	3	2	$2^{183.4}$	$2^{209.1}$	$2^{219.8}$	$2^{261.9}$	III
P_3	4	2	$2^{253.9}$	$2^{297.3}$	$2^{316.1}$	$2^{340.0}$	V

TABLE VI. COMPARISON OF PROPOSED PKE'S FEATURES WITH OTHER LATTICE-BASED SCHEMES

Features	[10]	[13]	[11]	[37]	Proposed Scheme
Underlying hard problem	MLWE	MLWR	LWE	MLWR	MLWR
Polynomial space	$\mathbb{Z}_q[x]/(x^{256}+1)$	$\mathbb{Z}_q[x]/(x^{256}+1)$	$\mathbb{Z}_q[x]$	$\mathbb{Z}_q[x]/(x^{256}+1)$	$\mathbb{Z}_q[x]/(x^{256}+1)$
Dimensions	1024	1024	640	1024	1024
Modulus	Prime: 3329	Power of 2 : 8192	Power of 2: 32768	Prime:8380417	Prime:7937
Analysis of polynomial multiplication methods	-	-	-	✓	✓
Polynomial multiplication method used	Incomplete NWC NTT	Karatsuba and ToomCook	Naive (schoolbook)	full NWC NTT	Incomplete NWC NTT
Sampling	Rejection Sampling	No need of rejection sampling	Inversion sampling	Rejection Sampling	Rejection Sampling
Secret distribution	Centered binomial distribution	Centered binomial distribution	Uniform distribution	Centered binomial distribution	Centered binomial distribution
Performance analysis	✓	✓	✓	✓	✓
Key sizes [bytes]	sk:1632, pk:800	sk:1568, pk:672	sk:19888, pk:9616	sk:1422, pk:1040	sk:3264, pk:1632
Avg. Execution time (μ s)	Enc:32131 Dec:48317.20	Enc:75, Dec:37.5	Enc:114038 Dec:126289	Enc:75.037 Dec:34.483	Enc:69.48 Dec:34.12
Avg. CPU cycles required	Enc:6870, Dec:962	Enc: 200 , Dec:100	Enc:14800 Dec:15970	Enc: 320 , Dec:200	Enc: 300 , Dec:120
BKW security	$2^{238.3}$	2^{292}	$2^{241.1}$	$2^{270.9}$	$2^{340.0}$

Note: ✓denotes "performed" and - denotes there is no result reported in the particular reference.

7937, the scheme maintains robust security levels while optimizing computational efficiency through the incomplete NTT.

D. Result and Comparison with the other Lattice-Based Schemes

The proposed scheme with high level security requirements has been implemented and tested on a Linux operating system. The CPU time required to execute the proposed key generation, encryption, and decryption functions is measured using Intel® VTune™ Profiler and is 172 μs, 69.48 μs, and 34.12 μs, respectively.

The outcomes of the proposed approach are compared to the other PKE candidates of the NIST's PQC project. NIST requested submission for PQC algorithms [2] in 2016 for the standardization process. After the internal reviews and public feedback, twenty-one LBC candidates were selected in the 1^{st} round. While in the 2^{nd} round only 9 algorithms were selected. Out of 9, 3 are high-speed as well as secure PQC algorithms named Frodo [11], Kyber [10], and Saber [13]. All these algorithms are based on lattice-based hard problems. Execution time of the proposed PKE is compared with these schemes.

Table VI presents a comparative analysis against various features of the proposed MLWR-based PKE scheme with other well-known LBC schemes, focusing on the underlying cryptographic problems, polynomial spaces, performance characteristics (such as execution time and key sizes), and security. Key differences include the polynomial multiplication methods used (e.g., NTT vs. Karatsuba), security analysis, and performance metrics such as execution time and CPU cycles.

The proposed scheme's public key and ciphertext sizes are both smaller than Frodo's scheme. Size of the secret key is smaller than the Frodo. Also, the average execution time and average CPU cycles required for the proposed scheme's encryption and decryption are comparable to that of the most efficient candidate, SABER. The average time needed to encrypt a 256-byte message is equivalent to that of SABER, whereas the average time required to decrypt a message is less than that of SABER.

The proposed scheme stands out for its high BKW security and relatively low execution times and CPU cycles, offering a superior balance of security, performance, and computational efficiency. This makes it a preferred choice for implementing post-quantum cryptographic protocols

VII. CONCLUSION

The proposed scheme attains a better performance compared to other lattice-based schemes. Also, this scheme provides prominent hardness and security, ingrained in the LWR variant. It has a faster and simpler encryption and decryption process. The key sizes are very compact. As the proposed scheme is based on a lattice-based hard problem, MLWR, it can be secure against quantum attacks.

Analysis highlighted the importance of selecting an appropriate modulus q to balance performance and security. Among the evaluated moduli, $q = 7937$ emerged as the most efficient

choice, providing a desirable trade-off between computational efficiency and security.

Proposed MLWR-based PKE scheme using incomplete NTT stands out due to its high security and efficient performance, making it an optimal choice for practical and secure post-quantum cryptographic applications.

Future work could explore further optimizations in parameter selection and the adaptation of the incomplete NTT methodology to other post-quantum cryptographic schemes, enhancing their applicability to a wider range of security-critical and resource-limited scenarios.

REFERENCES

[1] O. Regev, "An efficient quantum factoring algorithm," *J. ACM*, Dec. 2024, just Accepted. [Online]. Available: https://doi.org/10.1145/3708471

[2] G. Alagic, M. Bros, P. Ciadoux, D. Cooper, Q. Dang, T. Dang, J. Kelsey, J. Lichtinger, Y.-K. Liu, C. Miller *et al.*, *Status report on the fourth round of the nist post-quantum cryptography standardization process*. US Department of Commerce, National Institute of Standards and Technology, 2025.

[3] T. Jamil, "From bits to qubits: Comparative insights into classical and quantum computing systems," *International Journal of Advanced Computer Science & Applications*, vol. 16, no. 12, 2025.

[4] M. Ajtai, "Generating hard instances of lattice problems (extended abstract)," in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, ser. STOC '96. New York, NY, USA: Association for Computing Machinery, 1996, p. 99–108. [Online]. Available: https://doi.org/10.1145/237814.237838

[5] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *Journal of the ACM (JACM)*, vol. 56, no. 6, sep 2009. [Online]. Available: https://doi.org/10.1145/1568318.1568324

[6] A. Langlois, D. Stehlé, C. C. by Padro A Langlois, D. Stehlé, and A. Langlois, "Worst-case to average-case reductions for module lattices," *Designs, Codes and Cryptography 2014 75:3*, vol. 75, no. 3, pp. 565–599, feb 2014.

[7] A. Banerjee, C. Peikert, and A. Rosen, "Pseudorandom functions and lattices," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, vol. 7237 LNCS. Springer, 2012, pp. 719–737.

[8] P. Newton and S. Richelson, "A lower bound for proving hardness of learning with rounding with polynomial modulus." Berlin, Heidelberg: Springer-Verlag, 2023, p. 805–835. [Online]. Available: https://doi.org/10.1007/978-3-031-38554-4_26

[9] J. Alwen, S. Krenn, K. Pietrzak, and D. Wichs, "Learning with rounding, revisited," in *Advances in Cryptology – CRYPTO 2013*, R. Canetti and J. A. Garay, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 57–74.

[10] T. L. Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz and D. S. Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, "CRYSTALS-Kyber Algorithm Specifications and Supporting Documentation," *NIST PQC Round*, vol. 2, no. 4, pp. 1–32, 2021. [Online]. Available: https://pq-crystals.org/kyber/data/kyber-specification-round3-20210131.pdf

[11] L. Glabush, P. Longa, M. Naehrig, C. Peikert, D. Stebila, and F. Virdia, "FrodoKEM: A CCA-secure learning with errors key encapsulation mechanism," 2025. [Online]. Available: https://eprint.iacr.org/2025/1861

[12] J. Hoffstein, J. Pipher, and J. H. Silverman, "NTRU: A Ring-Based Public Key Cryptosystem," in *International Algorithmic Number Theory Symposium*. Springer, 1998, pp. 267–288.

[13] A. Karmakar, S. S. Roy, and F. Vercauteren, "SABER: Mod-LWR based KEM (Round 3 Submission)," *NIST PQC Round*, pp. 1–2, 2020. [Online]. Available: https://eprint.iacr.org/2018/230.pdf

[14] S. Bai, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé, "Crystals-dilithium: Algorithm specifications and supporting documentation (version 3.1)," *NIST Post-Quantum Cryptography Standardization Round*, vol. 3, 2021.

[15] Y. Li, J. Zhang, Z. Jin, and W. Qiao, "A multi-strategy improved horned lizard optimization algorithm and its application in engineering optimization," *International Journal of Computational Intelligence Systems*, vol. 18, 2025. [Online]. Available: https://api.semanticscholar.org/CorpusID:278251547

[16] H. Cheng, J. Großschädl, P. B. Rønne, and P. Y. Ryan, "Avrntru: Lightweight ntru-based post-quantum cryptography for 8-bit avr microcontrollers," pp. 1272–1277, 2021.

[17] D. J. Bernstein, B. B. Brumley, M.-S. Chen, C. Chuengsatiansup, T. Lange, A. Marotzke, B.-Y. Peng, N. Tuveri, C. van Vredendaal, and B.-Y. Yang, "Ntru prime: round-3 updates," *National Institute of Standards and Technology (NIST)*, 2020.

[18] Y. Sugizaki and D. Takahashi, "Improved implementation of number theoretic transform on nvidia gpu with tensor cores," in *Proceedings of the Supercomputing Asia and International Conference on High Performance Computing in Asia Pacific Region*, ser. SCA/HPCAsia '26. New York, NY, USA: Association for Computing Machinery, 2026, p. 142–152. [Online]. Available: https://doi.org/10.1145/3773656.3773673

[19] Z. Guitouni, N. Ammar, and M. Machhout, "An efficient hardware implementation of sha-3 using 3d cellular automata for secure wireless sensor networks," *International Journal of Information Security*, vol. 24, no. 3, p. 116, 2025.

[20] I. Prots' ko and A. Gryshchuk, "Implementing montgomery multiplication to speed-up the computation of modular exponentiation of multi-bit numbers," *Cybernetics and Systems Analysis*, vol. 60, no. 5, pp. 826–833, 2024.

[21] Y. Wang, Y. Zhao, and M. Wang, "On the Hardness of Ring/Module/Polynomial LWR Problems," *Cryptology ePrint Archive*, 2021. [Online]. Available: http://www.eprint.mirror.cypherpunks.su/2021/1026/1628236547.pdf

[22] T. Pöppelmann, T. Oder, and T. Güneysu, "High-performance ideal lattice-based cryptography on 8-bit atxmega microcontrollers," in *Progress in Cryptology – LATINCRYPT 2015*, K. Lauter and F. Rodríguez-Henríquez, Eds. Cham: Springer International Publishing, 2015, pp. 346–365.

[23] Y. Zhu, Z. Liu, and Y. Pan, "When NTT Meets Karatsuba: Preprocess-then-NTT Technique Revisited," in *International Conference on Information and Communications Security*, vol. 12919 LNCS. Springer, Cham, sep 2021, pp. 249–264.

[24] A. Satriawan, I. Syafalni, R. Mareta, I. Anshori, W. Shalannanda, and A. Barra, "Conceptual review on number theoretic transform and comprehensive review on its implementations," *IEEE Access*, vol. 11, pp. 70 288–70 316, 2023.

[25] H. Kondo, "Cooley-tukey fft over \mathbb {Q}_p via unramified cyclotomic extension," *arXiv preprint arXiv:2505.02509*, 2025.

[26] T. B. Rodrigues, A. Rodrigues, M. Goulão, P. Tomás, and L. Sousa, "Accelerating ntt with risc-v vector extension for fully homomorphic encryption," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2025, no. 4, pp. 711–736, 2025.

[27] A. Pouly and Y. Shen, "Solving the shortest vector problem in $2^{0.63269n+o(n)}$ time on random lattices," Cryptology ePrint Archive, Paper 2024/1805, 2024. [Online]. Available: https://eprint.iacr.org/2024/1805

[28] D. Micciancio and S. Goldwasser, *Complexity of lattice problems*, ser. The Springer International Series in Engineering and Computer Science. New York, NY: Springer, Oct. 2002, vol. 671.

[29] M. Ajtai and C. Dwork, "Public-key cryptosystem with worst-case/average-case equivalence," in *Conference Proceedings of the Annual ACM Symposium on Theory of Computing*. ACM, 1997, pp. 284–293.

[30] J. Sharafi and H. Daghigh, "A ring-lwe-based digital signature inspired by lindner–peikert scheme," *Journal of Mathematical Cryptology*, vol. 16, pp. 205 – 214, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:251369134

[31] P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, Z. Zhang *et al.*, "Falcon: Fast-fourier lattice-based compact signatures over ntru," *Submission to the NIST's post-quantum cryptography standardization process*, vol. 36, no. 5, pp. 1–75, 2018.

[32] K. Arai, "Method for mission analysis using tot-based prompt technology utilized generative ai satellite mission analysis for sagansat-0 of remote sensing satellite." *International Journal of Advanced Computer Science & Applications*, vol. 15, no. 9, 2024.

[33] Z. Zheng, A. Wang, and L. Qin, "Rejection Sampling Revisit: How to Choose Parameters in Lattice-Based Signature," *Mathematical Problems in Engineering*, vol. 2021, 2021.

[34] K. Boudgoust, C. Jeudy, A. Roux-Langlois, and W. Wen, "On the hardness of module learning with errors with short distributions: K. boudgoust et al." *Journal of Cryptology*, vol. 36, no. 1, p. 1, 2023.

[35] U. Banerjee, T. S. Ukyab, and A. P. Chandrakasan, "Sapphire: A configurable crypto-processor for post-quantum lattice-based protocols_2019," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2019, no. 4, p. 17–61, Aug. 2019. [Online]. Available: https://tches.iacr.org/index.php/TCHES/article/view/8344

[36] M. S. Turan, M. S. Turan, K. McKay, D. Chang, L. E. Bassham, J. Kang, N. D. Waller, J. M. Kelsey, and D. Hong, *Status report on the final round of the NIST lightweight cryptography standardization process*. US Department of Commerce, National Institute of Standards and Technology, 2023.

[37] A. A. Pandit and A. Mishra, "Efficient implementation of post quantum mlwr-based pke scheme using ntt," *Computers and Electrical Engineering*, vol. 118, p. 109358, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045790624002866

AUTHORS' PROFILE

Anupama Arjun Pandit (Student Member, IEEE) is currently pursuing Ph.D. degree in Computer Science and Engineering Department, at Defense Institute of Advanced Technology, Pune, India. She has completed her M. Tech. in CSE-IT from Walchand College of Engineering, Sangli, Maharashtra, India. Her research interest includes Cryptography and Post Quantum Cryptography (Code-based Cryptography and Lattice-based Cryptography). Her contribution to the current project is implementation, testing, analysis, and interpretation of the data, algorithm, and results.

Arun Mishra is working as an Associate Professor in Computer Science and Engineering Department at Defence Institute of Advanced Technology, Pune, India. He completed his Ph.D. from MNIT, Allahabad, India. He has more than 40 research publications in peer-reviewed journals & international conference proceedings. His areas of interest are Cryptography, Blockchain Technology, Post Quantum Cryptography (Code-based Cryptography, Lattice-based Cryptography and Hash-based Cryptography), Secure Software Engineering, and Secure Systems Design (isolation/ secure execution/ secure computations). In the current project, he contributed to the development of the concept and technique.