Compact and Efficient NTRU-based KEM with Scalable Ciphertext Compression

Zhichuang Liang
Department of Computer Science
Fudan University, Shanghai, China

Boyue Fang
Department of Computer Science
Fudan University, Shanghai, China

Jieyu Zheng
Department of Computer Science
Fudan University, Shanghai, China

Yunlei Zhao*
Department of Computer Science
Fudan University, Shanghai, China

ABSTRACT

Post-quantum cryptography (PQC) is critical to the next generation of network security. The NTRU lattice is a promising candidate to construct practical cryptosystems resistant to quantum computing attacks, and particularly plays a leading role in the ongoing NIST post-quantum cryptography standardization. On the one hand, it is benefited from a strong security guarantee since it has essentially not been broken over 24 years. On the other hand, all the known patent threats against NTRU have expired, which is deemed a critical factor for consideration when deploying PQC algorithms in reality. Nevertheless, there are still some obstacles to the computational efficiency and bandwidth complexity of NTRU-based constructions of key encapsulation mechanisms (KEM).

To address these issues, we propose a compact and efficient KEM based on the NTRU lattice, called CTRU, by introducing a scalable ciphertext compression technique. It demonstrates a new approach to decrypting NTRU ciphertext, where the plaintext message is recovered with the aid of our decoding algorithm in the scalable E_8 lattice (instead of eliminating the extra terms modulo p in traditional NTRU-based KEM schemes). The instantiation of CTRU is over the NTT-friendly rings of the form $\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$. We remark that the scalable ciphertext compression technique can also be applied to NTRU-based KEM schemes over other polynomial rings. In order to deal with the inconvenient issue that various NTT algorithms are needed for different n's, we present a unified NTT methodology over $\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$, $n \in \{512, 768, 1024\}$, such that only one type of NTT computation is required for different n's, which might be of independent interest.

To our knowledge, our CTRU is the most bandwidth efficient KEM based on the NTRU lattice up to now. In addition, roughly speaking, compared to other NTRU-based KEM schemes, CTRU has stronger security against known attacks, enjoys more robust CCA security reduction (starting from IND-CPA rather than OW-CPA), and its encapsulation and decapsulation processes are also among the most efficient. For example, when compared to the NIST Round 3 finalist NTRU-HRSS, our CTRU-768 has 15% smaller ciphertext size and its security is strengthened by $(45, 40)$ bits for classical and quantum security respectively. When compared to the NIST Round 3 finalist Kyber that is based on the Module-LWE (MLWE) assumption, CTRU has both smaller bandwidth and lower error probabilities at about the same security level.

1 INTRODUCTION

Most current public-key cryptographic schemes in use, which are based on the hardness assumptions of factoring large integers and solving (elliptic curve) discrete logarithms, will suffer from quantum attack, if practical quantum computers are built. These cryptosystems play an important role in ensuring the confidentiality and authenticity of communications on the Internet. With the increasing cryptographic security risks of quantum computing in recent years, post-quantum cryptography (PQC) has become a research focus for the crypto community. There are five main types of post-quantum cryptographic schemes: hash-based, code-based, lattice-based, multivariable-based, and isogeny-based schemes, among which lattice-based cryptography is commonly viewed as amongst the most promising one due to its outstanding balanced performance in security, communication bandwidth, and computational efficiency.

In the post-quantum cryptography standardization competition held by the U.S. National Institute of Standards and Technology (NIST), lattice-based schemes account for 26 out of 64 schemes in the first round [90], 12 out of 26 in the second round [91], and 7 out of 15 in the current third round [92]. Most of these lattice-based schemes are based on one of the following types: plain lattice and algebraically structured lattice (ideal lattice, NTRU lattice, and module lattice). They are mainly instantiated from the following two categories of hardness assumptions. The first category consists of *Learning With Errors* (LWE) [97] and its variants with algebraic structures such as *Ring-Learning With Errors* (RLWE) [84] and *Module-Learning With Errors* (MLWE) [79], as well as *Learning With Rounding* (LWR) [13] and its variants such as *Ring-Learning With Rounding* (RLWR) [13] and *Module-Learning With Rounding* (MLWR) [7]. The second category is the *NTRU* assumption [64].

NTRU, which stands for "$\underline{N}^{th}$*-Degree* $\underline{T}$*runcated Polynomial* $\underline{R}$*ing* $\underline{U}$*nits*", was first proposed by Jeffrey Hoffstein at the rump session Crypto96 [62], and it survived a lattice attack in 1997 [36]. With some improvements on security, NTRU was published by Hoffstein, Pipher and Silverman in 1998 [64], which is named NTRU-HPS in this work. NTRU-HPS was the first practical public key cryptosystem based on the lattice hardness assumptions over polynomial rings. There have been many variants of NTRU-HPS such as those proposed in [11, 19, 29, 52, 70, 85]. And NTRU has played a basic role in many cryptographic protocols, e.g., [46, 51, 55, 63, 80, 83]. In particular, NTRU-based schemes have achieved impressive success in the third round of NIST PQC standardization. Specifically, NTRU KEM (including NTRU-HRSS and NTRUEncrypt) [29] and Falcon

*Corresponding author: ylzhao@fudan.edu.cn.

signature scheme [51] are two of the seven finalists, and NTRU Prime KEM (including SNTRU Prime and NTRU LPRime) [19] is one of the alternate candidates in NIST PQC Round 3.

There are several reasons for using NTRU-based KEM schemes. The first is about its security. As the first practical lattice-based cryptographic scheme, until now NTRU-based KEM schemes have survived attacks and cryptanalysis over 24 years. Some efforts on provable security have been made in [101, 104, 107, 108]. But their resulting NTRU-based schemes are impractical since they have large parameters. In general, most current NTRU-based schemes remain unbroken. The second reason is that all the patent threats against NTRU have expired. However, there are some known patents that arguably threaten other lattice-based finalists such as Kyber [9] and Saber [14]. For example, besides more latent patent threats, U.S. patent 9094189 [54] threatens their "noisy Diffle-Hellman with reconciliation" structure, and U.S. patent 9246675 [42] threatens their decryption mechanisms. The patent threats are deemed a critical factor for consideration when deploying PQC standardized algorithms in reality. The third reason is that NTRU-based KEM schemes admit more flexible key sizes to be encapsulated (corresponding to the message space $\mathcal{M}$ in this work), varying according to the degree of the underlying quotient polynomial. In comparison, the KEM schemes based on MLWE and MLWR like Kyber and Saber encapsulate keys of fixed size that is restricted to the underlying quotient polynomial that is of degree 256 for Kyber and Saber.

On the other hand, currently there are also some drawbacks to NTRU-based KEM schemes. The first is about ciphertext compression. The importance of reducing ciphertext size is self-evident, since low communication bandwidth is friendly to internet protocols (e.g., TLS) and constrained devices in the internet of things (IoT). Though ciphertext compression is a quite mature technique for {R,M}LWE-based KEM schemes, it is at a very rusty stage for NTRU-based KEM constructions. Common NTRU-based encryption schemes [29, 48, 64, 101] consist of the ciphertext of the form $c = phr + m \bmod q$, where p is the message space modulus, h is the public key, r is the randomness, and m is the message to be encrypted. In the decryption process, one could compute $cf \bmod q = pgr + mf$, and clean out the term pgr via reduction modulo p. In order to obtain m, one can multiply the inverse of f modulo p, or directly reduce modulo p if $f = pf'+1$. It can be viewed as a unidimensional error-correction mechanism. However, if the ciphertext is further compressed, the error term in each component can not be eliminated via reduction modulo p and consequently the messages can not be recovered correctly. Without ciphertext compression, as a consequence, at about the same security level, the bandwidth of NTRU-based KEM schemes is usually larger than that of {R,M}LWE-based KEM schemes. The second drawback is about security reduction. For most NTRU-based KEM constructions, their chosen ciphertext attack (CCA) security is usually reduced to the one-way (OW-CPA) secure encryption instead of the traditional IND-CPA encryption. Above all, IND-CPA is a strictly stronger security notion than OW-CPA. OW-CPA can be transformed into IND-CPA, but at the price of further loosening the reduction bound particularly in the quantum random oracle model (QROM) [48]. One can also have a tight reduction from CCA security to OW-CPA *deterministic* public-key encryption (DPKE), but at the cost of a more complicated decapsulation process [19, 29]. More detailed discussions and clarifications on CCA security reduction of KEM in the ROM and the QROM are presented in Appendix A. As a consequence, it is still desirable for NTRU-based KEM constructions to have security reduction from CCA security to IND-CPA security, as is in {R,M}LWE-based KEM schemes.

1.1 Our Contributions

In this work, we present a new variant of NTRU-based cryptosystem, called CTRU, which can achieve scalable ciphertext compression and has CCA provable security reduced directly to IND-CPA. It consists of an IND-CPA secure public-key encryption, named CTRU.PKE, and an IND-CCA secure key encapsulation mechanism, named CTRU.KEM constructed through $\mathrm{FO}^{\not\perp}_{ID(pk),m}$ that is an enhanced variant of Fujisaki-Okamoto transformation [53, 65] with a short prefix of the public key into the hash function [47].

Our CTRU.PKE demonstrates a novel approach to constructing NTRU-based PKE. The description of CTRU is over the NTT-friendly rings of the form $\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$. We choose $n \in \{512, 768, 1024\}$ for NIST recommended security levels I, III and V, respectively, with the same $q = 3457$ set for all the three dimensions for ease of implementation simplicity and compatability. We recommend the case of $n = 768$ which could have a moderate post-quantum security and performance in reality.

1.1.1 New construction. The key generation algorithm in CTRU is similar to the exiting NTRU-based KEM schemes such as [19, 29, 48, 64]. CTRU uses $h = g/f$ as its public key and f as its secret key. We develop a new encryption algorithm which breaks through the limitation of ciphertext compression for NTRU-based KEM, such that we can compress the ciphertexts in the case of one single polynomial. To be specific, we encode every 4-bit messages into a scalable E_8 lattice point and hide its information by an RLWE instance, after which we compress the resulting ciphertext as many as possible. As for the decryption algorithm, we multiply the ciphertext polynomial by the secret polynomial, and finally recover the messages correctly with the aid of our decoding algorithm in the scalable E_8 lattice whenever the ℓ_2 norm of the error term is less than the sphere radius of the scalable E_8 lattice. An important point to note is that, different from most existing NTRU-based KEM schemes such as [29, 48, 64], in CTRU the message space modulus p is removed in the public key h and in the ciphertext c, as it is not needed there to recover the message m with our CTRU construction. The only reserved position for p is the secret key f, which has the form of $f = pf' + 1$. We show that the above steps constitute an IND-CPA secure PKE scheme: CTRU.PKE, based on the NTRU assumption and the RLWE assumption. Finally, we then apply the $\mathrm{FO}^{\not\perp}_{ID(pk),m}$ transformation [47] to get the IND-CCA secure CTRU.KEM.

1.1.2 Unified NTT. The NTT-based polynomial operations over $\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$ are very efficient. However, as the dimension n varies with CTRU, we have to equip with multiple NTT algorithms with different input/output lengths in accordance with each $n \in \{512, 768, 1024\}$. This brings inconvenient issues for software implementation and especially for hardware implementation. In this work, we overcome this problem by presenting the methodology of using a unified NTT technique to compute NTTs over $\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$ for all $n \in \{512, 768, 1024\}$ with $q = 3457$. Technically speaking, we split $f \in \mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$ into $\alpha \in \{2, 3, 4\}$ sub-polynomials of lower degrees, each of which is in $\mathbb{Z}_q[x]/(x^{256} - x^{128} + 1)$. We then design a 256-point unified NTT based on the ideas from [85, 88], and apply it to each sub-polynomial. Finally, their intermediate NTT results are combined to generate the final results. In this case, in order to obtain the public key (the quotient of two n-dimension polynomials), we need to compute the inversions in the rings of the form $\mathbb{Z}_q[x]/(x^{2\alpha} - \zeta)$, where ζ is some primitive root of unity in $\mathbb{Z}_q$. We use Cramer's Rule [57] to compute the inverse of polynomials of low degree.

1.1.3 Performance and comparisons. By careful evaluation and selection, we provide a set of parameters for CTRU, and present the recommended parameter sets in Section 5.1.3. Here, we make comparisons between CTRU on the recommended parameters and other prominent practical NTRU-based KEM schemes: NTRU-HRSS [29, 69], SNTRU Prime [19], NTTRU [85] and NTRU-C^{768}_{3457} [48], as well as Kyber [9]. The comparisons are summarized in Table 1.

To the best of our knowledge, CTRU is the first NTRU-based KEM scheme with scalable ciphertext compression via only one single ciphertext polynomial. From the comparisons, CTRU has the smallest bandwidth and the strongest security guarantees among all the practical NTRU-based KEM schemes. The error probabilities of CTRU are set according to the security level targeted by each set of parameters, which can be viewed as negligible in accordance with the security level. For example, when compared to the NIST Round 3 finalist NTRU-HRSS [29], our CTRU-768 has 15% smaller ciphertext size and its security is strengthened by $(45, 40)$ bits for classical and quantum security, respectively. When compared to the NIST Round 3 finalist Kyber [9] that is based on the MLWE assumption, the security of CTRU is slightly reduced for about 1 or 2 bits, due to the modulus $q = 3457$ in CTRU that is slightly larger than $q = 3329$ in Kyber. But roughly at the same level of security, CTRU has both smaller bandwidth and lower error probabilities. To the best of our knowledge, CTRU is the first NTRU-based KEM that enjoys all these advantages.

On negligible error vs. zero error. The error probability of CTRU-768 is set to be 2^{-187}, while that of NTRU-HRSS [29] is zero. Since the target security level of CTRU-768 is 164, the error probability of 2^{-187} is sufficiently low. In our opinion, it is quite paranoid that some NTRU-based KEM schemes, e.g., NTRU-HRSS, reduce the error probability to zero. One can see that the tradeoffs for no error vs. negligible error 2^{-187} are more than 40 bits of security and 15% smaller ciphertext size when compared to NTRU-HRSS. We also stress that we do not know how to have the well balance achieved by CTRU by simply adjusting parameters for the existing NTRU-based KEM schemes.

On security reduction. Our CTRU.PKE can achieve the IND-CPA security under the NTRU assumption and the RLWE assumption, while most of the existing practical NTRU-based PKEs only achieve OW-CPA security. The reduction advantage of CCA security of our CTRU.KEM is tighter than those of NTTRU [85] and NTRU-C^{768}_{3457} [48]. For example, in the quantum setting, the CCA reduction bound of CTRU.KEM is dominated by $O(\sqrt{q'\epsilon_{CPA}})$, while those of NTTRU and NTRU-C^{768}_{3457} are $O(q'\sqrt{\epsilon_{OW}})$ and $O(q'^{1.5}\sqrt[4]{\epsilon_{OW}})$ respectively, where $\epsilon_{CPA}(\epsilon_{OW})$ is the advantage against the underlying IND-CPA (resp., OW-CPA) secure PKE and q' is the total query number. However, NTRU-HRSS has a tight CCA reduction bound starting from OW-CPA *deterministic* PKE (DPKE), at the cost of more complicated and time-consuming decryption process [29]. In any case, IND-CPA is a strictly stronger security notion than OW-CPA.

1.1.4 Reference implementation and benchmark. We provide C reference implementation for CTRU-768, and perform benchmark comparisons with the related lattice-based KEM schemes (for those whose reference implementation codes are online available). The benchmark comparisons show that the encapsulation and decapsulation algorithms of CTRU-768 are among the most efficient. When compared to the reference implementation of NTRU-HRSS in NIST PQC Round 3, CTRU-768 is faster by 15X in KeyGen, 39X in Encaps, and 61X in Decaps, respectively. More details and discussions about the implementation and benchmark comparisons are referred to Section 7.

1.2 Related Work

In recent years, many NTRU variants have been proposed. Jarvis and Nevins [70] presented a new variant of NTRU-HPS [64] over the ring of Eisenstein integers $\mathbb{Z}[\omega]/(x^n - 1)$ where $\omega = e^{2\pi i/3}$, which has smaller key sizes and faster performance than NTRU-HPS. Bagheri et al. [11] generalized NTRU-HPS over bivariate polynomial rings of the form $(-1, -1)/(\mathbb{Z}[x, y]/(x^n - 1, y^n - 1))$ for stronger security and smaller public key sizes. Hülsing et al. [69] improved NTRU-HPS in terms of speed, key size, and ciphertext size, and presented NTRU-HRSS, which is one of the finalists in NIST PQC Round 3 now [29]. Bernstein et al. [18] proposed NTRU Prime, which aims for "an efficient implementation of high security prime-degree large-Galois-group inert-modulus ideal-lattice-based cryptography". It tweaks the textbook NTRU scheme to use some rings with less special structures, i.e., $\mathbb{Z}_q[x]/(x^n - x - 1)$, where both n and q are primes.

In order to obtain better performance of NTRU encryption, Lyubashevsky and Seiler [85] instantiated it over $\mathbb{Z}_{7681}[x]/(x^{768} - x^{384} + 1)$. Then Duman et al. [48] generalized the rings $\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$ with various n for flexible parameter selection. But all of them follow the similar structure of NTRU-HPS and do not support ciphertext compression.

Very recently, Fouque et al. [52] proposed a new NTRU variant named BAT. It shares many similarities with Falcon signature [51] where a trapdoor basis is required in the secret key, which makes its key generation complicated. BAT uses two linear equations in two unknowns to recover the secret and error, without introducing the modulus p to extract message. It reduces the ciphertext sizes by constructing its intermediate value as an RLWR instance (with binary secrets), and encrypts the message via ACWC_0 transformation [48]. However, ACWC_0 transformation consists of two terms, causing that there are some dozens of bytes in the second ciphertext. Another disadvantage is about the inflexibility of selecting parameters. Since BAT applies power-of-two cyclotomics $\mathbb{Z}_q[x]/(x^n + 1)$, it is inconvenient to find an underlying cyclotomic polynomial of some particular degree up to the next power of two. For example, BAT

Table 1: Comparisons between CTRU and other lattice-based KEM schemes. The column "Assumptions" refers to the underlying hardness assumptions. The column "Reduction" means that IND-CCA security is reduced to what kinds of CPA security, where "IND" ("OW") refers to indistinguishability (resp., one-wayness) and "RPKE" ("DPKE") refers to randomized (resp., deterministic) public-key encryptions. "Rings" refers to the underlying polynomial rings. The column "n" means the total dimension of algebraically structured lattices. "q" is the modulus. The public key sizes $|pk|$, ciphertext sizes $|ct|$, and B.W. (bandwidth, $|pk| + |ct|$) are measured in bytes. "Sec.C" and "Sec.Q" mean the estimated security expressed in bits in the classical and quantum setting respectively, which are gotten by the same methodology and scripts provided by Kyber and NTRU KEM in NIST PQC Round 3, where we minimize the target values if the two hardness problems, say NTRU and RLWE, have different security values. The column "δ" indicates the error probabilities.

Schemes	Assumptions	Reduction	Rings	n	q	$\|pk\|$	$\|ct\|$	B.W.	(Sec.C, Sec.Q)	δ
CTRU (Ours)	NTRU, RLWE	IND-CPA RPKE	$\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$	512	3457	768	640	1408	(118,107)	2^{-144}
				768	3457	1152	960	2112	(181,164)	2^{-187}
				1024	3457	1536	1408	2944	(255,231)	2^{-206}
NTRU-HRSS [29]	NTRU	OW-CPA DPKE	$\mathbb{Z}_q[x]/(x^n - 1)$	701	8192	1138	1138	2276	(136,124)	$2^{-\infty}$
SNTRU Prime-761 [19]	NTRU	OW-CPA DPKE	$\mathbb{Z}_q[x]/(x^n - x - 1)$	761	4591	1158	1039	2197	(153,137)	$2^{-\infty}$
NTTRU [85]	NTRU	OW-CPA RPKE	$\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$	768	7681	1248	1248	2496	(153,140)	2^{-1217}
NTRU-C_{3457}^{768} [48]	NTRU, RLWE	IND-CPA RPKE	$\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$	768	3457	1152	1184	2336	(171,155)	2^{-255}
Kyber [9]	MLWE	IND-CPA RPKE	$\mathbb{Z}_q[x]/(x^{n/k} + 1)$ $k = 2, 3, 4$	512	3329	800	768	1568	(118,107)	2^{-139}
				768	3329	1184	1088	2272	(183,166)	2^{-164}
				1024	3329	1568	1568	3136	(256,232)	2^{-174}

chooses $\mathbb{Z}_q[x]/(x^{512} + 1)$ and $\mathbb{Z}_q[x]/(x^{1024} + 1)$ for NIST recommended security levels I and V, but lacks of parameter set for level III, which, however, is the aimed and recommended security level for most lattice-based KEM schemes like Kyber [9] and our CTRU. Although BAT has an advantage of bandwidth, its key generation is 1,000 times slower than other NTRU-based KEM schemes, and there are some worries about its provable security based on the RLWR assumption with binary secrets which is quite a new assumption tailored for BAT. For the above reasons, we do not make a direct comparison between CTRU and BAT.

2 PRELIMINARIES

2.1 Notations and Definitions

Let $\mathbb{Z}$ and $\mathbb{R}$ be the set of rational integers and real numbers, respectively. Let n and q be some positive integers. Denote $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z} \cong \{0, 1, \ldots, q-1\}$ and $\mathbb{R}_q = \mathbb{R}/q\mathbb{R}$. Let $\mathbb{Z}_q^\times$ be the group of invertible elements of $\mathbb{Z}_q$. For any $x \in \mathbb{R}$, $\lfloor x \rceil$ denotes the closest integer to x. We denote $\mathbb{Z}[x]/(x^n - x^{n/2} + 1)$ and $\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$ by $\mathcal{R}$ and $\mathcal{R}_q$ respectively in this work. The elements in $\mathcal{R}$ or $\mathcal{R}_q$ are polynomials, which are denoted by regular font letters such as f, g. The polynomial, e.g., f, in $\mathcal{R}$ (or $\mathcal{R}_q$) can be represented in the form of power series: $f = \sum_{i=0}^{n-1} f_i x^i$, or in the form of vector: $f = (f_0, f_1, \ldots, f_{n-1})$, where $f_i \in \mathbb{Z}$ (or $f_i \in \mathbb{Z}_q$), $i = 0, 1, \ldots, n-1$. A function $\epsilon : \mathbb{N} \longrightarrow [0, 1]$ is negligible, if $\epsilon(\lambda) < 1/\lambda^c$ holds for any positive c and sufficiently large λ. Denote a negligible function by *negl*.

Cyclotomics. More details about cyclotomics can be found in [105]. Let m be a positive integer, $\xi_m = \exp(\frac{2\pi i}{m})$ be a m-th root of unity. The m-th cyclotomic polynomial $\Phi_m(x)$ is defined as $\Phi_m(x) = \prod_{j=1,\gcd(j,m)=1}^{m} (x - \xi_m^j)$. It is a monic irreducible polynomial of degree $\phi(m)$ in $\mathbb{Z}[x]$, where ϕ is the Euler function. The m-th cyclotomic field is $\mathbb{Q}(\xi_m) \cong \mathbb{Q}[x]/(\Phi_m(x))$ and its corresponding ring of integers is exactly $\mathbb{Z}[\xi_m] \cong \mathbb{Z}[x]/(\Phi_m(x))$. Most of cryptographic schemes based on algebraically structured lattices are defined over power-of-two cyclotomic rings, $\mathbb{Z}[x]/(x^n + 1)$ and $\mathbb{Z}_q[x]/(x^n + 1)$, where $n = 2^e$ is a power of two such that $x^n + 1$ is the 2^{e+1}-th cyclotomic polynomial. We use non-power-of-two cyclotomic rings $\mathbb{Z}[x]/(x^n - x^{n/2} + 1)$ and $\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$, where $n = 3^l \cdot 2^e, l \geq 0, e \geq 1$ throughout this paper and in this case $x^n - x^{n/2} + 1$ is the $3^{l+1} \cdot 2^e$-th cyclotomic polynomial.

Modular reductions. In this work, we expand the definition of modular reduction from $\mathbb{Z}$ to $\mathbb{R}$. For a positive number q, $r' = r \bmod {}^{\pm}q$ means that r' is the representative element of r in $[-\frac{q}{2}, \frac{q}{2})$. Let $r' = r \bmod q$ denote as the representative element of r in $[0, q)$.

Sizes of elements. Let q be a positive number. For any number $w \in \mathbb{R}$, denote by $\|w\|_{q,\infty} = |w \bmod {}^{\pm}q|$ its ℓ_∞ norm. If w is an n-dimension vector, then its ℓ_2 norm is defined as $\|w\|_{q,2} = \sqrt{\|w_0\|_{q,\infty}^2 + \cdots + \|w_{n-1}\|_{q,\infty}^2}$. Notice that $\|w\|_{q,2} = \|w\|_{q,\infty}$ holds for any number $w \in \mathbb{R}$.

Sets and Distributions. For a set D, we denote by $x \overset{\$}{\leftarrow} D$ sampling x from D uniformly at random. If D is a probability distribution, $x \leftarrow D$ means that x is chosen according to the distribution D. The centered binomial distribution B_η with respect to a positive integer η is defined as follows: Sample $(a_1, \ldots, a_\eta, b_1, \ldots, b_\eta) \overset{\$}{\leftarrow} \{0, 1\}^{2\eta}$, and output $\sum_{i=1}^{\eta} (a_i - b_i)$. Sampling a polynomial $f \leftarrow B_\eta$ means sampling each coefficient according to B_η individually.

2.2 Cryptographic Primitives

A public-key encryption scheme contains PKE = (KeyGen, Enc, Dec), with a message space $\mathcal{M}$. The key generation algorithm KeyGen returns a pair of public key and secret key (pk, sk). The encryption algorithm Enc takes a public key pk and a message $m \in \mathcal{M}$ to produce a ciphertext c. Denote by $\mathrm{Enc}(pk, m; coin)$ the encryption algorithm with an explicit randomness $coin$ if necessary. The deterministic decryption algorithm Dec takes a secret key sk and a ciphertext c, and outputs either a message $m \in \mathcal{M}$ or a special symbol $\perp$ to indicate a rejection. The decryption error δ of PKE is defined as $\mathrm{E}[\max_{m \in \mathcal{M}} \Pr[\mathrm{Dec}(sk, \mathrm{Enc}(pk, m))] \neq m] < \delta$. The advantage of an adversary A against *indistinguishability under chosen-plaintext attacks* (IND-CPA) for public-key encryption is defined as $\mathbf{Adv}^{\text{IND-CPA}}_{\text{PKE}}(\mathsf{A}) =$

$$\left| \Pr\left[b' = b : \begin{array}{c} (pk, sk) \leftarrow \mathrm{KeyGen}(); \\ (m_0, m_1, s) \leftarrow \mathsf{A}(pk); \\ b \xleftarrow{\$} \{0,1\}; c^* \leftarrow \mathrm{Enc}(pk, m_b); \\ b' \leftarrow \mathsf{A}(s, c^*) \end{array} \right] - \frac{1}{2} \right|.$$

A key encapsulation mechanism contains KEM = (KeyGen, Encaps, Decaps) with a key space $\mathcal{K}$. The key generation algorithm KeyGen returns a pair of public key and secret key (pk, sk). The encapsulation algorithm Encaps takes a public key pk to produce a ciphertext c and a key $K \in \mathcal{K}$. The deterministic decapsulation algorithm Decaps inputs a secret key sk and a ciphertext c, and outputs either a key $K \in \mathcal{K}$ or a special symbol $\perp$ indicating a rejection. The error probability δ of KEM is defined as $\Pr[\mathrm{Decaps}(sk, c) \neq K : (c, K) \leftarrow \mathrm{Encaps}(pk)] < \delta$. The advantage of an adversary A against *indistinguishability under chosen-ciphertext attacks* (IND-CCA) for KEM is defined as $\mathbf{Adv}^{\text{IND-CCA}}_{\text{KEM}}(\mathsf{A}) =$

$$\left| \Pr\left[b' = b : \begin{array}{c} (pk, sk) \leftarrow \mathrm{KeyGen}(); \\ b \xleftarrow{\$} \{0,1\}; \\ (c^*, K_0^*) \leftarrow \mathrm{Encaps}(pk); \\ K_1^* \xleftarrow{\$} \mathcal{K}; \\ b' \leftarrow \mathsf{A}^{\mathrm{Decaps}(\cdot)}(pk, c^*, K_b^*) \end{array} \right] - \frac{1}{2} \right|.$$

2.3 Hardness Assumptions

As the lattice cryptography evolved over the decades, the security of NTRU and its variants can be naturally viewed as two assumptions. One is the *NTRU* assumption [64], and the other is the *Ring-Learning with error* (RLWE) assumption [84], which are listed as follows. In some sense, the NTRU assumption can be viewed as a special case of the RLWE assumption. More details about NTRU cryptosystem and its applications can be seen in the excellent survey [102].

Definition 2.1 (NTRU assumption [64]). Let Ψ be a distribution over a polynomial ring R. Sample f and g according to Ψ, and f is invertible in R. Let $h = g/f$. The decisional NTRU assumption states that h is indistinguishable from a uniformly-random element in R. More precisely, the decisional NTRU assumption is hard if the advantage $\mathbf{Adv}^{\text{NTRU}}_{R,\Psi}(\mathsf{A})$ of any probabilistic polynomial time (PPT) adversary A is negligible, where $\mathbf{Adv}^{\text{NTRU}}_{R,\Psi}(\mathsf{A}) =$

$$\left| \Pr\left[b' = 1 : \begin{array}{c} f, g \leftarrow \Psi \wedge f^{-1} \in R \\ h = g/f \in R; b' \leftarrow \mathsf{A}(h) \end{array} \right] \right.$$
$$\left. - \Pr\left[b' = 1 : h \xleftarrow{\$} R; b' \leftarrow \mathsf{A}(h) \right] \right|.$$

Definition 2.2 (RLWE assumption [84]). Let Ψ be a distribution over a polynomial ring R. The (decisional) Ring-Learning with error (RLWE) assumption over R is to distinguish uniform samples $(h, c) \xleftarrow{\$} R \times R$ from samples $(h, c) \in R \times R$ where $h \xleftarrow{\$} R$ and $c = hr + e$ with $r, e \leftarrow \Psi$. It is hard if the advantage $\mathbf{Adv}^{\text{RLWE}}_{R,\Psi}(\mathsf{A})$ of any probabilistic polynomial time adversary A is negligible, where $\mathbf{Adv}^{\text{RLWE}}_{R,\Psi}(\mathsf{A}) =$

$$\left| \Pr\left[b' = 1 : \begin{array}{c} h \xleftarrow{\$} R; r, e \leftarrow \Psi; \\ c = hr + e \in R; b' \leftarrow \mathsf{A}(h, c) \end{array} \right] \right.$$
$$\left. - \Pr\left[b' = 1 : h \xleftarrow{\$} R; c \xleftarrow{\$} R; b' \leftarrow \mathsf{A}(h, c) \right] \right|.$$

3 THE LATTICE CODING

Before introducing our proposed NTRU-based KEM scheme, we present simple and efficient lattice coding algorithms. The motivation is that a dense lattice with efficient decoding algorithm is needed in our construction for better efficiency on recovering message and low enough error probability. The coding algorithms should satisfy the following conditions.

- The operations should be simple enough, and can be implemented by efficient arithmetic (better for integer-only operations).
- The decoding bound is large enough such that it leads to a high fault-tolerant mechanism.

We note that an 8-dimension lattice, named E_8 lattice (see [34], Chapter 4) could satisfy the above requirements to some extent. As for its density, there is a remarkable mathematical breakthrough that sphere packing in the E_8 lattice is proved to be optimal in the sense of the best density when packing in $\mathbb{R}^8$ [103]. As for the efficiency on coding, there has been simple executable encoding and decoding algorithms of the E_8 lattice in [33, 34]. However, the known coding algorithms in [33, 34] cannot be directly applied here. To work in our setting, we need to specify a one-to-one mapping from binary strings to the E_8 lattice points to encode messages. In this work, we specify such a mapping by choosing a basis for the scalable version of the E_8 lattice, which can transform the lattice points to the binary strings without involving Gaussian Elimination.

3.1 Coding with Scalable E_8 Lattice

The scalable E_8 lattice is constructed from the Extended Hamming Code with respect to dimension 8, which is defined as $H_8 = \{\mathbf{c} \in \{0,1\}^8 \mid \mathbf{c} = \mathbf{zH} \bmod 2, \mathbf{z} \in \{0,1\}^4\}$ where the binary matrix $\mathbf{H}$ is

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

Let $C = \{(x_1, x_1, x_2, x_2, x_3, x_3, x_4, x_4) \in \{0,1\}^8 \mid \sum x_i \equiv 0 \bmod 2\}$, where C is spanned by the up most three rows of $\mathbf{H}$. Then the scalable E_8 lattice (named E_8' lattice) is constructed as

$$\mathrm{E}_8' = \lambda \cdot [C \cup (C + \mathbf{c})] \subset [0, \lambda]^8$$

where $\mathbf{c} = (0, 1, 0, 1, 0, 1, 0, 1)$ is the last row of $\mathbf{H}$, $\lambda \in \mathbb{R}^+$ is the scale factor and $\lambda \cdot C$ means that all the elements in C multiply by λ.

3.1.1 Encoding algorithm of the E_8' lattice. The encoding algorithm of the E_8' lattice (see Algorithm 1) is to calculate $\lambda \cdot (\mathbf{kH} \bmod 2)$, given a 4-bit binary string $\mathbf{k}$, where $(\mathbf{kH} \bmod 2)$ can be computed efficiently by bitwise operations.

Algorithm 1 $\mathrm{Encode}_{E_8'}(\mathbf{k} \in \{0,1\}^4)$

```
v := $\lambda \cdot (\mathbf{kH} \bmod 2) \in [0, \lambda]^8$
return v
```

Algorithm 2 $\mathrm{Decode}_{E_8'}(\mathbf{x} = (x_0, \ldots, x_7) \in \mathbb{R}^8)$

```
1: Recall that $\mathbf{c} := (0, 1, 0, 1, 0, 1, 0, 1)$
2: $(\mathbf{k}_0, \mathsf{TotalCost}_0) := \mathrm{Decode}_{C'}(\mathbf{x})$
3: $(\mathbf{k}_1, \mathsf{TotalCost}_1) := \mathrm{Decode}_{C'}(\mathbf{x} - \lambda \cdot \mathbf{c})$
4: $b := \arg\min\{\mathsf{TotalCost}_0, \mathsf{TotalCost}_1\}$
5: $(k_0, k_1, k_2, k_3) := \mathbf{k}_b$
6: $\mathbf{k} := (k_0, k_1 \oplus k_0, k_3, b) \in \{0,1\}^4$
7: return k
```

Algorithm 3 $\mathrm{Decode}_{C'}(\mathbf{x} \in \mathbb{R}^8)$

```
$mind := +\infty$
$mini := 0$
$\mathsf{TotalCost} := 0$
for $i = 0 \ldots 3$ do
    $c_0 := \|x_{2i}\|^2_{2\lambda,2} + \|x_{2i+1}\|^2_{2\lambda,2}$
    $c_1 := \|x_{2i} - \lambda\|^2_{2\lambda,2} + \|x_{2i+1} - \lambda\|^2_{2\lambda,2}$
    $k_i := \arg\min\{c_0, c_1\}$
    $\mathsf{TotalCost} := \mathsf{TotalCost} + c_{k_i}$
    if $c_{1-k_i} - c_{k_i} < mind$ then
        $mind := c_{1-k_i} - c_{k_i}$
        $mini := i$
    end if
end for
if $k_0 + k_1 + k_2 + k_3 \bmod 2 = 1$ then
    $k_{mini} := 1 - k_{mini}$
    $\mathsf{TotalCost} := \mathsf{TotalCost} + mind$
end if
$\mathbf{k} := (k_0, k_1, k_2, k_3) \in \{0,1\}^4$
return $(\mathbf{k}, \mathsf{TotalCost})$
```

3.1.2 Decoding algorithm. Given any $\mathbf{x} \in \mathbb{R}^8$, the decoding algorithm is to find the solution of the closest vector problem (CVP) of $\mathbf{x}$ in the E_8' lattice, which is denoted by $\lambda \cdot \mathbf{k}'\mathbf{H} \bmod 2$, and it outputs the 4-bit string $\mathbf{k}'$. To solve the CVP of $\mathbf{x} \in \mathbb{R}^8$ in the E_8' lattice, we turn to solve the CVP of $\mathbf{x}$ and $\mathbf{x} - \lambda\mathbf{c}$ in the lattice $C' = \lambda \cdot C$. The one that has smaller distance is the final answer.

We briefly introduce the idea of solving the CVP in the lattice C' here. Given $\mathbf{x} \in \mathbb{R}^8$, for every two components in $\mathbf{x}$, determine whether they are close to $(0, 0)$ or (λ, λ). Assign the corresponding component of $\mathbf{k}$ to 0 if the former is true, and 1 otherwise. If $\sum k_i \bmod 2 = 0$ holds, it indicates that $\lambda \cdot (k_0, k_0, k_1, k_1, k_2, k_2, k_3, k_3)$ is the solution. However, $\sum k_i \bmod 2$ might be equal to 1. Then we choose the secondly closest vector, $\lambda \cdot (k_0', k_0', k_1', k_1', k_2', k_2', k_3', k_3')$, where there will be at most one-bit difference between (k_0, k_1, k_2, k_3) and (k_0', k_1', k_2', k_3'). The detailed algorithm is given in Algorithm 2, along with Algorithm 3 as its subroutines. Note that in Algorithm 3, *mind* and *mini* are set to store the minimal difference of the components and the corresponding index, respectively.

Finally, $\mathrm{Decode}_{C'}$ in Algorithm 2 will output the 4-bit string (k_0, k_1, k_2, k_3) such that the lattice point $\lambda \cdot (k_0, k_0 \oplus b, k_1, k_1 \oplus b, k_2, k_2 \oplus b, k_3, k_3 \oplus b)$ is closest to $\mathbf{x}$ in the E_8' lattice. Since the lattice point has the form of $\lambda \cdot (\mathbf{kH} \bmod 2)$, the decoding result $\mathbf{k}$ can be obtained by tweaking the solution of the CVP in the E_8' lattice, as in line 5 and line 6 in Algorithm 2.

3.2 Bound of Correct Decoding

Theorem 3.1 gives a bound of correct decoding w.r.t. Algorithm 2. Briefly speaking, for any 8-dimension vector which is close enough to the given E_8' lattice point under the metric of ℓ_2 norm, it can be decoded into the same 4-bit string that generates the lattice point. This theorem is helpful when we try to recover the targeted message from the given lattice point with error terms in our schemes.

THEOREM 3.1 (CORRECTNESS BOUND OF THE E_8' LATTICE DECODING). *For any given $\mathbf{k}_1 \in \{0,1\}^4$, denote $\mathbf{v}_1 := Encode_{E_8'}(\mathbf{k}_1)$. For any $\mathbf{v}_2 \in \mathbb{R}^8$, denote $\mathbf{k}_2 := Decode_{E_8'}(\mathbf{v}_2)$. If $\|\mathbf{v}_2 - \mathbf{v}_1\|_{2\lambda,2} < \lambda$, then $\mathbf{k}_1 = \mathbf{k}_2$.*

PROOF. According to the construction of the Extended Hamming Code H_8, we know that its minimal Hamming distance is 4. Thus, the radius of sphere packing in the E_8' lattice we used is $\frac{1}{2}\sqrt{4 \cdot \lambda^2} = \lambda$. As shown in Algorithm 1, $\mathbf{v}_1$ is the lattice point generated from $\mathbf{k}_1$. As for $\mathbf{v}_2 \in \mathbb{R}^8$, if $\|\mathbf{v}_2 - \mathbf{v}_1\|_{2\lambda,2} < \lambda$, the solution of the CVP about $\mathbf{v}_2$ in the E_8' lattice is $\mathbf{v}_1$. Since $\mathrm{Decode}_{E_8'}$ in Algorithm 2 will output the 4-bit string finally, instead of the intermediate solution of the CVP, $\mathbf{v}_1$ is also generated from $\mathbf{k}_2$, i.e., $\mathbf{v}_1 = \lambda \cdot (\mathbf{k}_2\mathbf{H} \bmod 2)$, which indicates that $\mathbf{k}_1 = \mathbf{k}_2$. □

4 CTRU: CONSTRUCTION AND ANALYSIS

In this section, we propose our new cryptosystem based on NTRU lattice, named CTRU, which contains an IND-CPA secure public-key encryption (CTRU.PKE) and an IND-CCA secure key encapsulation mechanism (CTRU.KEM). CTRU has a similar form of public key and secret key to those of the traditional NTRU-based KEM schemes, but the method to recover message in CTRU is significantly different from them. With our construction, CTRU will achieve smaller ciphertext sizes with scalable ciphertext compression.

4.1 Proposal Description

Our CTRU.PKE scheme is specified in Algorithm 4-6. Restate that $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$, where n and q are the ring parameters. Let q_2 be the ciphertext modulus, which is usually set to be a power of two. Let p be the message space modulus, satisfying $\gcd(q, p) = 1$. We fix $p = 2$ in this work. Let Ψ be the distribution over $\mathcal{R}$. For presentation simplicity, the secret terms, f', g, r and e, are all taken from Ψ. In general, they can taken from different distributions. Let $\mathcal{M} = \{0, 1\}^{n/2}$ denote the message space, where each $m \in \mathcal{M}$ can be seen as a $\frac{n}{2}$-dimension polynomial with coefficients in $\{0, 1\}$.

Algorithm 4 CTRU.PKE.KeyGen()

```
$f', g \leftarrow \Psi$
$f := pf' + 1$
If $f$ is not invertible in $\mathcal{R}_q$, restart.
$h := g/f$
**return** $(pk := h, sk := f)$
```

Algorithm 5 CTRU.PKE.Enc($pk = h$, $m \in \mathcal{M}$)

```
$r, e \leftarrow \Psi$
$\sigma := hr + e$
$c := \left\lfloor \frac{q_2}{q}(\sigma + \lfloor \text{PolyEncode}(m) \rceil) \right\rceil \bmod q_2$
**return** $c$
```

Algorithm 6 CTRU.PKE.Dec($sk = f$, c)

```
$m := \text{PolyDecode}\left(cf \bmod {}^{\pm} q_2\right)$
**return** $m$
```

Algorithm 7 PolyEncode($m = \sum_{i=0}^{n/2-1} m_i x^i \in \mathcal{M}$)

```
$\mathrm{E}_8' := \frac{q}{2} \cdot [C \cup (C + \mathbf{c})] \subset [0, \frac{q}{2}]^8$
**for** $i = 0 \ldots n/8 - 1$ **do**
    $\mathbf{k}_i := (m_{4i}, m_{4i+1}, m_{4i+2}, m_{4i+3}) \in \{0, 1\}^4$
    $(v_{8i}, v_{8i+1}, \ldots, v_{8i+7}) := \text{Encode}_{E_8'}(\mathbf{k}_i) \in [0, \frac{q}{2}]^8$
**end for**
$v := \sum_{i=0}^{n-1} v_i x^i$
**return** $v$
```

Algorithm 8 PolyDecode($v = \sum_{i=0}^{n-1} v_i x^i \in \mathcal{R}_{q_2}$)

```
$\mathrm{E}_8'' := \frac{q_2}{2} \cdot [C \cup (C + \mathbf{c})] \subset [0, \frac{q_2}{2}]^8$
**for** $i = 0 \ldots n/8 - 1$ **do**
    $\mathbf{x}_i := (v_{8i}, v_{8i+1}, \ldots, v_{8i+7}) \in \mathbb{R}^8$
    $(m_{4i}, m_{4i+1}, m_{4i+2}, m_{4i+3}) := \text{Decode}_{E_8''}(\mathbf{x}_i) \in \{0, 1\}^4$
**end for**
$m := \sum_{i=0}^{n/2-1} m_i x^i \in \mathcal{M}$
**return** $m$
```

The PolyEncode algorithm and PolyDecode algorithm are described in Algorithm 7 and 8, respectively. Specifically, we construct the E_8' lattice with the scale factor $\frac{q}{2}$ in Algorithm 7. That is, the encoding algorithm works over $\mathrm{E}_8' := \frac{q}{2} \cdot [C \cup (C + \mathbf{c})]$. The PolyEncode algorithm splits each $m \in \mathcal{M}$ into some quadruples, each of which will be encoded via $\text{Encode}_{E_8'}$. As for PolyDecode algorithm, the decoding algorithm works over the lattice $\mathrm{E}_8'' := \frac{q_2}{2} \cdot [C \cup (C + \mathbf{c})]$. It splits $v \in \mathcal{R}_{q_2}$ into some octets, each of which will be decoded via $\text{Decode}_{E_8''}$. The final message m can be recovered by combining all the 4-bit binary strings output by $\text{Decode}_{E_8''}$.

We construct our CTRU.KEM=(Keygen, Encaps, Decaps) by applying $\text{FO}^{\not\perp}_{ID(pk),m}$, a variant of Fujisaki-Okamoto (FO) transformation [53, 65] aimed for the strengthened IND-CCA security in multi-user setting [47]. Let ι, γ be positive integers. We prefer to choose $\iota, \gamma \geq 256$ for strong security. Let $\mathcal{H} : \{0, 1\}^* \to \mathcal{K} \times \mathcal{COINS}$ be a hash function, where $\mathcal{K}$ is the shared key space of CTRU.KEM and $\mathcal{COINS}$ is the randomness space of CTRU.PKE.Enc. Note that we make explicit the randomness in CTRU.PKE.Enc here. Define $\mathcal{H}_1(\cdot)$ as $\mathcal{H}(\cdot)$'s partial output that is mapped into $\mathcal{K}$. Let $\mathcal{PK}$ be the public key space of CTRU.PKE. Let $ID : \mathcal{PK} \to \{0, 1\}^{\gamma}$ be a fixed-output length function. The algorithms of CTRU.KEM are described in Algorithm 9-11.

Algorithm 9 CTRU.KEM.KeyGen()

```
$(pk, sk) \leftarrow$ CTRU.PKE.KeyGen()
$z \xleftarrow{\$} \{0, 1\}^{\iota}$
**return** $(pk' := pk, sk' := (sk, z))$
```

Algorithm 10 CTUR.KEM.Encaps(pk)

```
$m \xleftarrow{\$} \mathcal{M}$
$(K, coin) := \mathcal{H}(ID(pk), m)$
$c :=$ CTRU.PKE.Enc$(pk, m; coin)$
**return** $(c, K)$
```

Algorithm 11 CTRU.KEM.Decaps((sk, z), c)

```
$m' :=$ CTRU.PKE.Dec$(sk, c)$
$(K', coin') := \mathcal{H}(ID(pk), m')$
$\tilde{K} := \mathcal{H}_1(ID(pk), z, c)$
**if** $m' \neq \perp$ and $c =$ CTRU.PKE.Enc$(pk, m'; coin')$ **then**
    **return** $K'$
**else**
    **return** $\tilde{K}$
**end if**
```

4.2 Correctness Analysis

Lemma 4.1. *It holds that* $cf \bmod {}^{\pm} q_2 = \frac{q_2}{q}((\frac{q}{q_2}c)f \bmod {}^{\pm} q)$.

Proof. Since polynomial multiplication can be described as matrix-vector multiplication, which keeps the linearity, it holds that $(\frac{q}{q_2}c)f = \frac{q}{q_2}(cf)$. There exits an integral vector $\theta \in \mathbb{Z}^n$ such that $\frac{q}{q_2}cf \bmod {}^{\pm} q = \frac{q}{q_2}cf + q\theta$ and $-\frac{q}{2} \leq \frac{q}{q_2}cf + q\theta < \frac{q}{2}$. Thus, we have $-\frac{q_2}{2} \leq cf + q_2\theta < \frac{q_2}{2}$. Hence, we obtain

$$cf \bmod {}^{\pm} q_2 = cf + q_2\theta = \frac{q_2}{q}(\frac{q}{q_2}cf + q\theta) = \frac{q_2}{q}((\frac{q}{q_2}c)f \bmod {}^{\pm} q).$$

□

THEOREM 4.2. *Let Ψ be the distribution over the ring $\mathcal{R}$, and q, q_2, p be positive integers. Let $f', g, r, e \leftarrow \Psi$. Let $\varepsilon_1 \leftarrow \chi_1$, where χ_1 is the distribution over $\mathcal{R}$ defined as follows: Sample $s \overset{\$}{\leftarrow} \mathcal{R}_2$ and output $\left(\lfloor \frac{q}{2}s \rceil - \frac{q}{2}s\right) \bmod {}^{\pm}q$. And, let $\varepsilon_2 \leftarrow \chi_2$, where χ_2 is the distribution over $\mathcal{R}$ defined as follows: Sample $\sigma \overset{\$}{\leftarrow} \mathcal{R}_q$ and $s \overset{\$}{\leftarrow} \mathcal{R}_2$ and output $\left[\left\lfloor \frac{q_2}{q}(\sigma + \lfloor \frac{q}{2}s \rceil)\right\rceil - \frac{q_2}{q}(\sigma + \lfloor \frac{q}{2}s \rceil)\right] \bmod {}^{\pm}q_2$. Let Err_i be the i-th octet of $gr + ef + (\varepsilon_1 + \frac{q}{q_2}\varepsilon_2)f$. Denote $1 - \delta = Pr\left[\|Err_i\|_{q,2} < \frac{q}{2}\right]$. Then, the error probability of CTRU is δ.*

PROOF. Scale the E_8'' lattice and $cf \bmod {}^{\pm}q_2$ by the factor q/q_2. According to Lemma 4.1, we have

$$\begin{aligned} m &= \mathrm{PolyDecode}_{E_8''}\left(cf \bmod {}^{\pm}q_2\right) \\ &= \mathrm{PolyDecode}_{E_8'}\left((\frac{q}{q_2}c)f \bmod {}^{\pm}q\right) \end{aligned} \tag{1}$$

in Algorithm 6. Since $m \overset{\$}{\leftarrow} \mathcal{M}$ in Algorithm 10, the result of PolyEncode(m) in Algorithm 5 can be denoted by $\frac{q}{2}s$ where $s \overset{\$}{\leftarrow} \mathcal{R}_2$. Based on the hardness of the NTRU assumption and the RLWE assumption, σ in line 2 in Algorithm 5 is pseudo-random in $\mathcal{R}_q$. Therefore, the value of c in line 3 in Algorithm 5 is

$$c = \left\lfloor \frac{q_2}{q}(\sigma + \lfloor \frac{q}{2}s \rceil)\right\rceil \bmod q_2 = \frac{q_2}{q}(\sigma + \frac{q}{2}s + \varepsilon_1) + \varepsilon_2 \bmod q_2.$$

With $\sigma = hr + e$, $h = g/f$ and $f = 2f' + 1$, for the formula (1) we get

$$\begin{aligned} (\frac{q}{q_2}c)f \bmod {}^{\pm}q &= \frac{q}{q_2}[\frac{q_2}{q}(\sigma + \frac{q}{2}s + \varepsilon_1) + \varepsilon_2] \cdot f \bmod {}^{\pm}q \\ &= \frac{q}{2}s(2f' + 1) + \sigma f + (\varepsilon_1 + \frac{q}{q_2}\varepsilon_2)f \bmod {}^{\pm}q \\ &= \frac{q}{2}s + gr + ef + (\varepsilon_1 + \frac{q}{q_2}\varepsilon_2)f \bmod {}^{\pm}q \end{aligned} \tag{2}$$

Each octet of $\frac{q}{2}s$ in (2) is essentially a lattice point in the E_8' lattice, which we denoted by $\frac{q}{2}(\mathbf{k}_i\mathbf{H} \bmod 2)$. From Theorem 3.1 we know that to recover $\mathbf{k}_i$, it should hold $\|\mathrm{Err}_i\|_{q,2} < \frac{q}{2}$, where Err_i is the i-th octet of $gr + ef + (\varepsilon_1 + \frac{q}{q_2}\varepsilon_2)f$. □

The form of polynomial product in the ring $\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$ is presented in detail in Appendix B. The error probability is estimated by using a Python script. The results for the selected parameters are given in Table 2.

4.3 Provable Security

We prove that CTRU.PKE is IND-CPA secure under the NTRU assumption and the RLWE assumption.

THEOREM 4.3 (IND-CPA SECURITY). *For any adversary* A*, there exits adversaries* B *and* C *such that $\boldsymbol{Adv}^{\text{IND-CPA}}_{\text{CTRU.PKE}}(\mathsf{A}) \le \boldsymbol{Adv}^{\text{NTRU}}_{\mathcal{R}_q,\Psi}(\mathsf{B}) + \boldsymbol{Adv}^{\text{RLWE}}_{\mathcal{R}_q,\Psi}(\mathsf{C})$.*

PROOF. We complete our proof through a sequence of games $\mathbf{G}_0$, $\mathbf{G}_1$ and $\mathbf{G}_2$. Let A be the adversary against the IND-CPA security experiment. Denote by $\mathbf{Succ}_i$ the event that A wins in the game $\mathbf{G}_i$, that is, A outputs b' such that $b' = b$ in $\mathbf{G}_i$.

Game $\mathbf{G}_0$. This game is the original IND-CPA security experiment. Thus, $\mathbf{Adv}^{\text{IND-CPA}}_{\text{CTRU.PKE}}(\mathsf{A}) = |\Pr[\mathbf{Succ}_0] - 1/2|$.

Game $\mathbf{G}_1$. This game is the same as $\mathbf{G}_0$, except that replacing the public key $h = g/f$ in the KeyGen by $h \overset{\$}{\leftarrow} \mathcal{R}_q$. To distinguish $\mathbf{G}_1$ from $\mathbf{G}_0$ is equivalent to solve an NTRU problem. More precisely, there exits an adversary B with the same running time as that of A such that $|\Pr[\mathbf{Succ}_0] - \Pr[\mathbf{Succ}_1]| \le \mathbf{Adv}^{\text{NTRU}}_{\mathcal{R}_q,\Psi}(\mathsf{B})$.

Game $\mathbf{G}_2$. This game is the same as $\mathbf{G}_1$, except that using uniformly random elements from $\mathcal{R}_q$ to replace σ in the encryption. Similarly, there exits an adversary C with the same running time as that of A such that $|\Pr[\mathbf{Succ}_1] - \Pr[\mathbf{Succ}_2]| \le \mathbf{Adv}^{\text{RLWE}}_{\mathcal{R}_q,\Psi}(\mathsf{C})$.

In Game $\mathbf{G}_2$, for any given m_b, according to Algorithm 5 and 7, m_b is split into $n/8$ quadruples. Denote the i-th quadruple of m_b as $m_b^{(i)}$, which will later be operated to output the i-th octet of the ciphertext c that is denoted as $c^{(i)}$, $i = 0, 1, \ldots, n/8 - 1$. Since $c^{(i)}$ is only dependent on $m_b^{(i)}$ and other parts of m_b do not interfere with $c^{(i)}$, our aim is to prove that $c^{(i)}$ is independent of $m_b^{(i)}$, $i = 0, 1, \ldots, n/8 - 1$. For any i and any given $m_b^{(i)}$, $\lfloor \mathrm{Encode}_{E_8'}(m_b^{(i)}) \rceil$ is fixed. Based on the uniform randomness of σ in $\mathcal{R}_q$, its i-th octet (denoted as $\sigma^{(i)}$) is uniformly random in $\mathbb{Z}_q^8$, so is $\sigma^{(i)} + \lfloor \mathrm{Encode}_{E_8'}(m_b^{(i)}) \rceil$. Therefore, the resulting $c^{(i)}$ is subject to the distribution $\lfloor \frac{q_2}{q}u \rceil \bmod q_2$ where u is uniformly random in $\mathbb{Z}_q^8$, which implies that $c^{(i)}$ is independent of $m_b^{(i)}$. Hence, each $c^{(i)}$ leaks no information of the corresponding $m_b^{(i)}$, $i = 0, 1, \ldots, n/8 - 1$. We have $\Pr[\mathbf{Succ}_2] = 1/2$.

Combining all the probabilities finishes the proof. □

By applying the $\mathrm{FO}^{\not\perp}_{ID(pk),m}$ transformation and adapting the results given in [47], we have the following results on CCA security of CTRU in the random oracle model (ROM) [15] and quantum random oracle model (QROM) [25].

THEOREM 4.4 (IND-CCA SECURITY IN THE ROM AND QROM [47]). *Let ℓ be the min-entropy [53] of $ID(pk)$, i.e., $\ell = H_\infty(ID(pk))$, where $(pk, sk) \leftarrow$ CTRU.PKE.KeyGen. For any (quantum) adversary* A*, making at most q_D decapsulation queries, q_H (Q)RO queries, against the IND-CCA security of CTRU.KEM, there exits a (quantum) adversary* B *with roughly the same running time of* A*, such that:*

- *In the ROM, it holds that $\boldsymbol{Adv}^{\text{IND-CCA}}_{\text{CTRU.KEM}}(\mathsf{A}) \le$*

$$2\left(\boldsymbol{Adv}^{\text{IND-CPA}}_{\text{CTRU.PKE}}(\mathsf{B}) + \frac{q_H + 1}{|\mathcal{M}|}\right) + \frac{q_H}{2^{\iota}} + (q_H + q_D)\delta + \frac{1}{2^{\ell}};$$

- *In the QROM, it holds that $\boldsymbol{Adv}^{\text{IND-CCA}}_{\text{CTRU.KEM}}(\mathsf{A}) \le$*

$$2\sqrt{q_{HD}\boldsymbol{Adv}^{\text{IND-CPA}}_{\text{CTRU.PKE}}(\mathsf{B})} + \frac{4q_{HD}}{\sqrt{|\mathcal{M}|}} + \frac{4(q_H + 1)}{\sqrt{2^{\iota}}} + 16q_{HD}^2\delta + \frac{1}{|\mathcal{M}|} + \frac{1}{2^{\ell}},$$

where $q_{HD} := q_H + q_D + 1$.

The detailed discussions and clarifications on CCA security reduction of KEM in the ROM and the QROM are given in Appendix A.

4.4 Discussions and Comparisons

The rings. As in [48, 85], we choose non-power-of-two cyclotomics $\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$ with respect to $n = 3^l \cdot 2^e$ and prime q, This type of ring allows very fast NTT-based polynomial multiplication if the ring moduli are set to be NTT-friendly. Moreover, it also allows very flexible parameter selection, since there are many integral n of the form $3^l \cdot 2^e, l \geq 0, e \geq 1$.

The message modulus. Note that the modulus p is removed in the public key h (i.e., $h = g/f$) and in the ciphertext c of our CTRU, for the reason that p is not needed in h and c to recover the message m in our construction. The only reserved position of p is the secret key f, which has the form of $f = pf' + 1$. Since $\gcd(q, p) = 1$ is required for NTRU-based KEM schemes, we can use $p = 2$ instead of $p = 3$. A smaller p can lead to a lower error probability. Note that for other NTRU-based KEM schemes with power-of-two modulus q as in NTRU-HRSS [29], p is set to be 3 because it is the smallest integer co-prime to the power-of-2 modulus.

The decryption mechanism. Technically speaking, the ciphertext of NTRU-based PKE schemes [29, 48, 64, 101] has the form of $c = phr + m \bmod q$. One can recover the message m through a unidimensional error-correction mechanism, after computing $cf \bmod q$. Instead, we use a multi-dimension coding mechanism. We encode each 4-bit messages into a lattice point in the E_8' lattice. They can be recovered correctly with the aid of the E_8' decoding algorithm if the ℓ_2 norm of the error term is less than the sphere radius of the E_8' lattice.

The ciphertext compression. To the best of our knowledge, CTRU is the first NTRU-based KEM with scalable ciphertext compression via a single polynomial. The ciphertext modulus q_2 is adjustable, depending on the bits to be dropped. The reason that most NTRU-based KEM schemes fail to compress ciphertext is that the message cannot be recovered via reduction modulo p once the ciphertext is compressed.

5 CONCRETE HARDNESS AND PARAMETER SELECTION

In this section, we first estimate and select parameters for CTRU, by applying the methodology of core-SVP hardness estimation [6]. Then, we present the refined gate-count estimate, by using the scripts provided by Kyber and NTRU Prime in NIST PQC Round 3. Finally, we overview and discuss some recent attacks beyond the core-SVP hardness.

5.1 Parameter Selection with Core-SVP

5.1.1 Primal attack and dual attack. Currently, for the parameters selected for most practical lattice-based cryptosystems, the dominant attacks considered are the lattice-based primal and dual attacks. The primal attack is to solve the *unique-Short Vector Problem* (u-SVP) in the lattice by constructing an integer *embedding lattice* (Kannan embedding [75], Bai-Galbraith embedding [12], etc). The most common lattice reduction algorithm is the BKZ algorithm [30, 99]. Given a lattice basis, the *blocksize*, which we denote by b, is necessarily chosen to recover the short vector while running the BKZ algorithm. NTRU problem can be treated as a u-SVP instance in the NTRU lattice [36], while a u-SVP instance can also be constructed from the LWE problem. The dual attack [87] is to solve the *decisional* LWE problem, consisting of using the BKZ algorithm in the dual lattice, so as to recover part of the secret and infer the final secret vector.

5.1.2 Core-SVP hardness of CTRU. Following the simple and conservative methodology of the core-SVP hardness developed from [6], the best known cost of running SVP solver on b-dimension sublattice is $2^{0.292b}$ for the classical case and $2^{0.265b}$ for the quantum case. These cost models can be used for conservative estimates of the security of our schemes. Note that the number of samples is set to be $2n$ for NTRU problem (resp., n for LWE problem), since the adversary is given such samples. We estimate the classical and quantum core-SVP hardness security of CTRU via the Python script from [6, 9, 26]. The concrete results are given in Table 2.

5.1.3 Parameter sets. The parameter sets of CTRU are given in Table 2, where those in red are the recommended parameters also given in Table 1. Though the parameters in red are marked as recommended, we believe the other parameter sets are still very useful in certain application scenarios. Note that in Table 1 we did not list the security against the dual attack. The reason is that the dual attack was considered less realistic than the primal attack, and was not taken for concrete hardness estimates in many lattice-based cryptosystems including Kyber in NIST PQC Round 3 [9]. For ease of a fair comparison, the security estimate against the dual attack was not listed in Table 1.

The ring dimension n is chosen from $\{512, 768, 1024\}$, corresponding to the targeted security levels I, III and V recommended by NIST. We stress that selecting these n's for CTRU is only for simplicity. The ring modulus q is set to 3457, and q_2 is the ciphertext modulus. Recall that we fix the message space modulus $p = 2$ and the underlying cyclotomic polynomial $\Phi(x) = x^n - x^{n/2} + 1$, which are omitted in Table 2. Ψ is the probability distribution which is set to be the centered binomial distribution B_2 or B_3. The public key sizes $|pk|$, ciphertext sizes $|ct|$ and B.W. (bandwidth, $|pk| + |ct|$) are measured in terms of bytes. "Sec.C" and "Sec.Q" mean the estimated security level expressed in bits in the classical and quantum settings respectively, where all the types of NTRU attack, LWE primal attack, and LWE dual attack are considered. The last column "δ" indicates the error probability, which is evaluated by a script according to the analysis given in Section 4.2.

5.2 Refined Gate-Count Estimate

As for the quantum gates and space complexity related to the LWE problem, we use the same gate number estimation method as Kyber, NTRU KEM, and SNTRU Prime in NIST PQC Round 3. Briefly speaking, it uses the probabilistic simulation of [40] rather than the GSA-intersect model of [5, 6] to determine the BKZ blocksize b for a successful attack. And it relies on the concrete estimation for the cost of sieving in gates from [4]. It also accounts for the "few dimensions for free" proposed in [45], which permits to solve SVP in dimension b by sieving in a somewhat smaller dimension $b_0 = b - O(b)$. Finally, it dismisses the dual attack as realistically more expensive than the primal attack. In particular, in the dual attack, exploiting the short vectors generated by the Nearest Neighbor Search used in lattice sieving is not compatible with the "dimension

Table 2: Parameter sets of CTRU

Schemes	n	q	q_2	Ψ	$\|pk\|$	$\|ct\|$	B.W.	NTRU (Sec.C, Sec.Q)	LWE, primal (Sec.C, Sec.Q)	LWE, dual (Sec.C, Sec.Q)	δ
CTRU-512	512	3457	2^9	B_2	768	576	1344	(111,100)	(111,100)	(110,100)	2^{-120}
	512	3457	2^{10}	B_3	768	640	1408	(118,107)	(118,107)	(117,106)	2^{-144}
CTRU-768	768	3457	2^{10}	B_2	1152	960	2112	(181,164)	(181,164)	(180,163)	2^{-187}
	768	3457	3457	B_3	1152	1152	2304	(192,174)	(192,174)	(190,173)	2^{-143}
	768	3457	2^{11}	B_3	1152	1056	2208	(192,174)	(192,174)	(190,173)	2^{-125}
CTRU-1024	1024	3457	2^{11}	B_2	1536	1408	2944	(255,231)	(255,231)	(252,229)	2^{-206}
	1024	3457	2^{10}	B_2	1536	1280	2816	(255,231)	(255,231)	(252,229)	2^{-137}
	1024	3457	3457	B_3	1536	1536	3072	(269,244)	(269,244)	(266,241)	2^{-104}

Table 3: Gate-count estimate of CTRU parameters. d is the optimal lattice dimension for the attack. b is the BKZ block-size. b' is the sieving dimension accounting for "dimensions for free". Gates and memory are expressed in bits. The last column means the required log(gates) values by NIST.

Schemes	Ψ	d	b	b'	log(gates)	log(memory)	log(gates) by NIST
CTRU-512	B_2	1007	386	350	144.1	88.4	143
	B_3	1025	411	373	150.9	93.3	
CTRU-768	B_2	1467	634	583	214.2	137.9	207
	B_3	1498	671	618	224.6	145.3	
CTRU-1024	B_2	1919	890	825	286.1	188.9	272
	B_3	1958	939	871	299.7	198.5	

for free" trick [45]. The scripts for these refined estimates are provided in a git branch of the leaky-LWE estimator [40][1]. The results of CTRU parameter sets are shown in Table 3. It is estimated in [9] that the actual cost may not be more than 16 bits away from this estimate in either direction.

5.3 Attacks Beyond Core-SVP Hardness

5.3.1 Hybrid attack. The works [61, 91, 92] consider the hybrid attack as the most powerful against NTRU-based cryptosystems. However, even with many heuristic and theoretical analysis on hybrid attack [27, 61, 67, 106], so far it still fails to make significant security impact on NTRU-based cryptosystems partially due to the memory constraints. By improving the collision attack on NTRU problem, it is suggested in [89] that the mixed attack complexity estimate used for NTRU problem is unreliable, and there are both overestimation and underestimation. Judging from the current hybrid and meet-in-the-middle (MITM) attacks on NTRU problem, there is an estimation bias in the security estimates of NTRU-based KEMs, but this bias does not make a big difference to the claimed security. For example, under the MITM search, the security of NTRU KEM in NIST PQC Round 3 may be 2^{-8} less than the acclaimed value in the worst situation [89].

5.3.2 Recent advances on dual attack. There are some recent progress on the dual attack, and we discuss their impacts on CTRU. Duc et al. [44] propose that fast Fourier transform (FFT) can be useful to the dual attack. As for the small coefficients of the secrets, various improvements can also be achieved [1, 27, 31]. Albrecht and Martin [1] propose a re-randomization and smaller-dimensional lattice reduction method, and investigate the method for generating coefficients of short vectors in the dual attack. Guo and Thomas [58] show that the current security estimates from the primal attacks are overestimated. Espitau et al. [50] achieve a dual attack that outperforms the primal attack. These attacks can be combined with the hybrid attack proposed in [68] to achieve a further optimized attack under specific parameters [27, 77, 100]. Very recently, MATZOV [86] further optimizes the dual attack, and claims that the impact of its methods is larger than those of Guo and Thomas's work [58]. It is also mentioned in [86] that the newly developed methods might also be applicable to NTRU-based cryptosystems (e.g., by improving the hybrid attack). The improvements of dual attacks mentioned above have potential threats to the security of CTRU (as well as to other cryptosystems based on algebraically structured lattices). This line of research is still actively ongoing, and there is still no mature and convincing estimate method up to now.

5.3.3 S-unit attack. The basis of the S-unit attack is the unit attack: finding a short generator. On the basis of the constant-degree algorithm proposed in [49, 60], Biasse et al. [21] present a quantum polynomial time algorithm, which is the basis for generating the generator used in the unit attack and S-unit attack. Then, the unit attack is to shorten the generator by reducing the modulus of the unit, and the idea is based on the variant of the LLL algorithm [32] to reduce the size of the generator in the S-unit group. That is, it replaces y_i with y_i/ϵ, thereby reducing the size of y_i, where y_i refers to the size of the generator and ϵ is the reduction factor of the modulus of the unit. The S-unit attack is briefly recalled in Appendix C. Campbell et al. [28] consider the application of the cycloid structure to the unit attack, which mainly depends on the simple generator of the cycloid unit. Under the cycloid structure, the determinant is easy to determine, and is larger than the logarithmic length of the private key, which means that the private key can be recovered through the LLL algorithm.

After establishing a set of short vectors, the simple reduction repeatedly uses $v - u$ to replace v, thereby reducing the modulus of vector v, where u belongs to the set of short vectors. This idea is discovered in [10, 32]. The difference is that the algorithm proposed by Avanzi and Howard [10] can be applied to any lattice, but is limited to the ℓ_2 norm, while the algorithm proposed by Cohen [32]

[1] https://github.com/lducas/leaky-LWE-Estimator/tree/NIST-round3

is applicable to more norms. Pellet-Mary et al. [94] analyze the algorithm of Avanzi and Howard [10], and apply it to S-unit. They point out that the S-unit attack could achieve shorter vectors than existing methods, but still with exponential time for an exponentially large approximation factor. Very recently, Bernstein and Tanja [20] further improve the S-unit attack.

Up to now, it is still an open problem to predict the effectiveness of the reduction inside the unit attacks. The statistical experiments on various m'-th cyclotomics (with respect to power-of-two m') show that the efficiency of the S-unit attacks is much higher than a spherical model of the same lattice for $m' \in \{128, 256, 512\}$ [17]. The effect is about a factor of 2^{-3}, 2^{-6} and 2^{-11}, respectively. Therefore, even with a conservative estimate, the security impact on CTRU may not exceed a factor of 2^{-11}.

5.3.4 BKW attack. For cryptographic schemes to which the BKW method can be applied, the combined methods proposed in [2, 23, 59, 76], which extend the BKW method, can be the most efficient method for specific parameters. These methods require a large number of samples, and their security estimates are based on the analysis of lattice basis reduction, either by solving the encoding problem in the lattice or by converting to a u-SVP problem [3, 81, 82]. These attacks do not affect the security of CTRU, because the parameters chosen for CTRU do not meet the conditions of BKW.

5.3.5 Side channel attack. Ravi et al. [96] construct some ciphertexts with specific structures where the key information exists in the intermediate variables, so as to recover the key through side channel attack (SCA). They apply this attack to NTRU KEM and NTRU Prime in NIST PQC Round 3, which can recover the full secret keys through a few thousands of chosen ciphertext queries. This type of SCA-aided chosen ciphertext attack is not directly applicable to CTRU, but might be possible to be improved against CTRU.

5.3.6 Other attacks. Algebraic attacks [21, 28, 38, 39] and dense sublattice attacks [77] also provide new ideas for LWE-based cryptographic analysis. However, these attacks do not currently affect the acclaimed security of the proposed CTRU parameters.

6 POLYNOMIAL OPERATIONS IN CTRU

From a computational point of view, the fundamental and also time-consuming operations in NTRU-based schemes are the multiplications and divisions of the elements in the rings $\mathbb{Z}_q[x]/(\Phi(x))$. Number theoretic transform (NTT) is a special case of fast Fourier transform (FFT) over a finite field [95]. NTT is the most efficient method for computing polynomial multiplication of high degrees, due to its quasilinear complexity $O(n \log n)$. The complete NTT-based multiplication with respect to f and g is $INTT(NTT(f) \circ NTT(g))$, where NTT is the forward transform, $INTT$ is the inverse transform and "$\circ$" is the point-wise multiplication.

The FFT trick [16] is a fast algorithm to compute NTT, via the Chinese Remainder Theorem (CRT) in the ring form. Briefly speaking, given two co-prime polynomials g and h, the CRT isomorphism is that $\varphi : \mathbb{Z}_q[x]/(gh) \cong \mathbb{Z}_q[x]/(g) \times \mathbb{Z}_q[x]/(h)$ along with $\varphi(f) = (f \bmod g, f \bmod h)$. In the case of the radix-2 FFT trick, given $g = x^m - \zeta$ and $h = x^m + \zeta$, where ζ is invertible in $\mathbb{Z}_q$, the computation of the forward FFT tirck and inverse FFT tirck can be conducted via Cooley-Tukey butterfly [35] and Gentleman-Sande butterfly [56], respectively. The former indicates the computation from (f_i, f_j) to $(f_i + \zeta \cdot f_j, f_i - \zeta \cdot f_j)$, while the later indicates the computation from (f'_i, f'_j) to $(f'_i + f'_j, (f'_i - f'_j) \cdot \zeta^{-1})$.

$$\begin{tikzcd}
\mathbb{Z}_q[x]/(x^{\alpha\cdot N} - x^{\alpha\cdot N/2} + 1) \arrow[r, "\varphi_1"] \arrow[d, dashed, "n\text{-point NTT}"] & \left(\mathbb{Z}_q[y]/(y^N - y^{N/2} + 1)\right)[x]/(x^\alpha - y) \arrow[d, "N\text{-point NTT}"] \\
\prod_{k=0}^{N/2^\beta - 1} \mathbb{Z}_q[x]/(x^{\alpha\cdot 2^\beta} - \zeta^{\tau(k)}) & \left(\prod_{k=0}^{N/2^\beta - 1} \mathbb{Z}_q[y]/(y^{2^\beta} - \zeta^{\tau(k)})\right)[x]/(x^\alpha - y) \arrow[l, "\varphi_2"']
\end{tikzcd}$$

Figure 1: Map road for unified NTT

6.1 Unified NTT

In this work, we consider $n = \alpha \cdot N$, where $\alpha \in \{2, 3, 4\}$ is called the splitting-parameter and N is a power of two. In fact, α can be chosen more freely as arbitrary values of the form $2^i 3^j, i \geq 0, j \geq 0$. With the traditional NTT technique, when the dimension n changes we need to use different NTT algorithms of various input/output lengths to compute polynomial multiplications over $\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$. This causes much inconvenience to software and particularly hardware implementations. To address this issue, we unify the various n-point NTTs through an N-point NTT, which is referred to as the unified NTT technique. For $n \in \{512, 768, 1024\}$, we fix $N = 256$ and choose $\alpha \in \{2, 3, 4\}$. With this technique, we only focus on the implementation of the N-point NTT, which serves as the unified procedure to be invoked for different n's. Specifically, the computation of NTT over $\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$ is divided into three steps. For presentation simplicity, we only give the procedures of the forward transform as follows, since the inverse transform can be obtained by inverting these procedures. The map road is shown in Figure 1.

Step 1. Construct a splitting-polynomial map φ_1 :

$$\mathbb{Z}_q[x]/(x^{\alpha\cdot N} - x^{\alpha\cdot N/2} + 1) \to \left(\mathbb{Z}_q[y]/(y^N - y^{N/2} + 1)\right)[x]/(x^\alpha - y)$$

$$f = \sum_{i=0}^{\alpha\cdot N-1} f_i x^i \mapsto \sum_{j=0}^{\alpha-1} F_j x^j$$

where $F_j = \sum_{i=0}^{N-1} f_{\alpha\cdot i+j} y^i \in \mathbb{Z}_q[y]/(y^N - y^{N/2} + 1)$. Namely, the n-dimension polynomial is split into α N-dimension sub-polynomials.

Step 2. Apply the unified N-point NTT to F_j over $\mathbb{Z}_q[y]/(y^N - y^{N/2} + 1)$, $j = 0, 1, \ldots, \alpha - 1$. Specifically, inspired by NTTRU [85], there is a mapping such that $\mathbb{Z}_q[y]/(y^N - y^{N/2} + 1) \cong \mathbb{Z}_q[y]/(y^{N/2} - \zeta_1) \times \mathbb{Z}_q[y]/(y^{N/2} - \zeta_2)$ where $\zeta_1 + \zeta_2 = 1$ and $\zeta_1 \cdot \zeta_2 = 1$. Let q be the prime satisfying $\frac{3N}{2^\beta} | (q - 1)$, where $\beta \in \mathbb{N}$ is called the truncating-parameter, such that it exits the primitive $\frac{3N}{2^\beta}$-th root of unity ζ in $\mathbb{Z}_q$. To apply the radix-2 FFT trick, we choose $\zeta_1 = \zeta^{N/2^{\beta+1}} \bmod q$ and $\zeta_2 = \zeta_1^5 = \zeta^{5N/2^{\beta+1}} \bmod q$. Thus, both $y^{N/2} - \zeta_1$ and $y^{N/2} - \zeta_2$ can be recursively split down into degree-2^β terms like $y^{2^\beta} \pm \zeta$. The idea of truncating FFT trick originates from [88]. Therefore, $\mathbb{Z}_q[y]/(y^N - y^{N/2} + 1)$ can be decomposed into $\prod_{k=0}^{N/2^{\beta}-1} \mathbb{Z}_q[y]/(y^{2^{\beta}} - \zeta^{\tau(k)})$, where $\tau(k)$ is the power of ζ of the k-th term and we start the index k from zero. Let $\hat{F}_j$ be the NTT result of F_j and $\hat{F}_{j,l}$ be its l-th coefficient, $l = 0, 1, \ldots, N-1$. Hence, we can write

$$\hat{F}_j = (\sum_{l=0}^{2^{\beta}-1} \hat{F}_{j,l}\, y^l, \sum_{l=0}^{2^{\beta}-1} \hat{F}_{j,l+2^{\beta}}\, y^l, \ldots, \sum_{l=0}^{2^{\beta}-1} \hat{F}_{j,l+N-2^{\beta}}\, y^l)$$
$$\in \prod_{k=0}^{N/2^{\beta}-1} \mathbb{Z}_q[y]/(y^{2^{\beta}} - \zeta^{\tau(k)})$$

Step 3. Combine the intermediate values and obtain the final result by the map φ_2 :

$$\left(\prod_{k=0}^{N/2^{\beta}-1} \mathbb{Z}_q[y]/(y^{2^{\beta}} - \zeta^{\tau(k)})\right)[x]/(x^{\alpha} - y) \rightarrow \prod_{k=0}^{N/2^{\beta}-1} \mathbb{Z}_q[x]/(x^{\alpha \cdot 2^{\beta}} - \zeta^{\tau(k)})$$
$$\sum_{j=0}^{\alpha-1} \hat{F}_j x^j \mapsto \hat{f}$$

where $\hat{f} = \sum_{i=0}^{\alpha \cdot N-1} \hat{f}_i x^i$ is the NTT result of f. Its i-th coefficient is $\hat{f}_i = \hat{F}_{j,l}$, where $j = i \bmod \alpha$ and $l = \lfloor \frac{i}{\alpha} \rfloor$. It can be rewritten as:

$$\hat{f} = (\sum_{i=0}^{\alpha \cdot 2^{\beta}-1} \hat{f}_i x^i, \sum_{i=0}^{\alpha \cdot 2^{\beta}-1} \hat{f}_{i+\alpha \cdot 2^{\beta}} x^i, \ldots, \sum_{i=0}^{\alpha \cdot 2^{\beta}-1} \hat{f}_{i+n-\alpha \cdot 2^{\beta}} x^i)$$
$$\in \prod_{k=0}^{N/2^{\beta}-1} \mathbb{Z}_q[x]/(x^{\alpha \cdot 2^{\beta}} - \zeta^{\tau(k)}) \quad (3)$$

In this work, we choose $\beta = 1$ and $q = 3457$, where the primitive 384-th root of unity $\zeta = 55$ exits in $\mathbb{Z}_{3457}$. In this case, the point-wise multiplication is the corresponding 2α-dimension polynomial multiplication in $\mathbb{Z}_q[x]/(x^{2\alpha} - \zeta^{\tau(k)})$, $\alpha \in \{2, 3, 4\}$, $k = 0, 1, \ldots, N/2-1$.

6.2 Base Case Inversion

Utilizing the NTT techniques to compute the public key $h = g/f$ is essentially to compute $h = INTT(\hat{g} \circ \hat{f}^{-1})$. Here, $\hat{g} = NTT(g)$ and $\hat{f} = NTT(f)$ are of the form (3), and $\hat{f}^{-1}$ is gotten by computing a series of the inverses of 2α-dimension sub-polynomials (with respect to $\hat{f}$) in $\mathbb{Z}_q[x]/(x^{2\alpha} - \zeta^{\tau(k)})$, $\alpha \in \{2, 3, 4\}$, $k = 0, 1, \ldots, N/2-1$. The inverse of the elements in $\mathbb{Z}_q[x]/(x^{2\alpha} - \zeta^{\tau(k)})$ can be computed by Cramer's Rule [57].

Take $\mathbb{Z}_q[x]/(x^4 - \zeta)$ as an example. Let f be a degree-3 polynomial in $\mathbb{Z}_q[x]/(x^4 - \zeta)$, and denote its inverse by f', which implies $f \cdot f' = 1 \bmod x^4 - \zeta$. It can be written in the form of matrix-vector multiplication:

$$\begin{bmatrix} f_0 & \zeta f_3 & \zeta f_2 & \zeta f_1 \\ f_1 & f_0 & \zeta f_3 & \zeta f_2 \\ f_2 & f_1 & f_0 & \zeta f_3 \\ f_3 & f_2 & f_1 & f_0 \end{bmatrix} \cdot \begin{bmatrix} f'_0 \\ f'_1 \\ f'_2 \\ f'_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \quad (4)$$

Let Δ be the determinant of the coefficient matrix. Hence, the inverse of f exits if and only if $\Delta \neq 0$. In this case, according to Cramer's Rule, there is a unique f', whose individual components are given by

$$f'_i = \frac{\Delta_i}{\Delta}, i = 0, 1, 2, 3 \quad (5)$$

where Δ_i is the determinant of the matrix generated by replacing the $(i+1)$-th column of the coefficient matrix with $(1, 0, 0, 0)^T$. And Δ^{-1} can be computed by using Fermat's Little Theorem, i.e., $\Delta^{-1} \equiv \Delta^{q-2} \bmod q$.

7 IMPLEMENTATION AND BENCHMARK

We provide the portable C implementation of our CTRU for the recommended parameter set of $(n = 768, q = 3457, q_2 = 2^{10}, \Psi = B_2)$. As for the prefix $ID(pk)$ of the public key h in CTRU, we use the first 33 bytes of the bit-packed NTT representation of h. It is reasonable, since h is computationally indistinguishable from a uniformly random polynomial in $\mathcal{R}_q$ and the forward NTT transform keeps the randomness property (i.e., h is random, so is $NTT(h)$). Assuming uniformly random h, the first 22 coefficients have the min-entropy of more than 256 bits and occupy 33 bytes in the bit-packed NTT representation since each coefficient has 12 bits.

All the benchmark tests are run on an Intel(R) Core(TM) i7-10510U CPU at 2.3GHz (16 GB memory) with Turbo Boost and Hyperthreading disabled. The operating system is Ubuntu 20.04 LTS with Linux Kernel 4.4.0 and the gcc version is 9.4.0. The compiler flag is listed as follows: *-Wall -march=native -mtune=native -O3 -fomit-frame-pointer -Wno-unknown-pragmas*. We run the corresponding KEM algorithms for 10,000 times and calculate the average CPU cycles. The source codes of NTRU-HRSS, SNTRU-Prime and Kyber are taken from their Round 3 supporting documentations or their websites, while those of NTTRU are taken from [85]. However, the FO transformation in Kyber has been changed to $\text{FO}^{\not\perp}_{ID(pk),m}$ as in [47], since it is the fastest implementation of Kyber. One regret is that the source codes of NTRU-C^{768}_{3457} are not online available in [48], so we omit its results. The benchmark results of those schemes are shown in Table 4. For the sake of completeness, we also provide the comparison between CTRU and Saber [14] in Appendix D.

From Table 4, we can see that the encapsulation (Encaps) and decapsulation (Decaps) processes of CTRU are among the most efficient. When compared to NTRU-HRSS and SNTRU Prime-761, the efficiency improvements of CTRU-768 are benefited from the applications of NTT in polynomial operations. Note that CTRU-768 is faster than NTRU-HRSS by 15X in KeyGen, 39X in Encaps, and 61X in Decaps, respectively. The Decaps of CTRU is slightly slower than that of NTTRU, on the following grounds: (1) the decoding algorithm of the E'_8 lattice costs extra time; (2) NTT is invalid in $\mathcal{R}_{q_2}$ w.r.t. power-of-two q_2 in the decryption process, so we turn to schoolbook algorithm instead. The KeyGen of CTRU-768 needs to compute the inverse of degree-5 polynomials, whereas NTTRU's larger modulus $q = 7681$ allows simpler degree-2 polynomial inversions. However, with the smaller $q = 3457$, CTRU-768 has shorter pubic key size (7.6% shorter than that of NTTRU) and stronger security (164-bit quantum security of CTRU-768 vs. 140-bit security of NTTRU). If necessary, CTRU can choose parameter sets w.r.t. $q = 7681$ to obtain a more efficient KeyGen process comparable to that of NTTRU, which also further accelerates the processes of the Encaps and Decaps, but at the cost of bandwidth or security. Note that in practice the KeyGen is run once and for all, and its computational cost is less sensitive to most cryptographic applications. When compared to Kyber-768, the efficiency improvements in the Encaps and Decaps of CTRU-768 are mainly due to the fact that there is only one polynomial multiplication in CTRU-768's encryption process (which is also re-run with the Decaps), whereas there is a complicated polynomial matrix-vector multiplication in Kyber-768's encryption process.

Table 4: CPU cycles of lattice-based schemes (in kilo cycles).

Schemes	KeyGen	Encaps	Decaps
CTRU-768	8.2×10^3	80.9	151.8
NTRU-HRSS [29]	127.6×10^3	3.2×10^3	9.4×10^3
SNTRU Prime-761 [19]	17.1×10^3	9.0×10^3	23.7×10^3
NTTRU [85]	157.4	98.9	142.4
Kyber-768 [9]	140.3	159.0	205.9

REFERENCES

[1] Martin R. Albrecht. 2017. On Dual Lattice Attacks Against Small-Secret LWE and Parameter Choices in HElib and SEAL. In *EUROCRYPT 2017*, Vol. 10211. 103–129.

[2] Martin R. Albrecht, Carlos Cid, Jean-Charles Faugère, Robert Fitzpatrick, and Ludovic Perret. 2015. On the complexity of the BKW algorithm on LWE. *Des. Codes Cryptogr.* 74, 2 (2015), 325–354.

[3] Martin R. Albrecht, Robert Fitzpatrick, and Florian Göpfert. 2013. On the Efficacy of Solving LWE by Reduction to Unique-SVP. In *ICISC 2013*, Vol. 8565. 293–310.

[4] Martin R. Albrecht, Vlad Gheorghiu, Eamonn W. Postlethwaite, and John M. Schanck. 2020. Estimating Quantum Speedups for Lattice Sieves. In *ASIACRYPT 2020*, Vol. 12492. 583–613.

[5] Martin R. Albrecht, Florian Göpfert, Fernando Virdia, and Thomas Wunderer. 2017. Revisiting the Expected Cost of Solving uSVP and Applications to LWE. In *ASIACRYPT 2017*, Vol. 10624. 297–322.

[6] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. 2016. Post-quantum Key Exchange - A New Hope. In *USENIX 2016*. 327–343.

[7] Jacob Alperin-Sheriff and Daniel Apon. 2016. Dimension-Preserving Reductions from LWE to LWR. *IACR Cryptol. ePrint Arch.* (2016), 589.

[8] Andris Ambainis, Mike Hamburg, and Dominique Unruh. 2019. Quantum Security Proofs Using Semi-classical Oracles. In *CRYPTO 2019*, Vol. 11693. 269–295.

[9] Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. 2020. CYRYSTALS-Kyber - algorithm specifications and supporting documentation (version 3.01). *NIST Post-Quantum Cryptography Standardization Process* (2020).

[10] Roberto Avanzi and Howard M. Heys (Eds.). 2017. *SAC 2016 - 23rd International Conference, St. John's, NL, Canada, August 10-12, 2016, Revised Selected Papers.* Vol. 10532.

[11] Khadijeh Bagheri, Mohammad-Reza Sadeghi, and Daniel Panario. 2018. A non-commutative cryptosystem based on quaternion algebras. *Des. Codes Cryptogr.* 86, 10 (2018), 2345–2377.

[12] Shi Bai and Steven D. Galbraith. 2014. Lattice Decoding Attacks on Binary LWE. In *ACISP 2014*, Vol. 8544. 322–337.

[13] Abhishek Banerjee, Chris Peikert, and Alon Rosen. 2012. Pseudorandom Functions and Lattices. In *EUROCRYPT 2012*, Vol. 7237. 719–737.

[14] Andrea Basso, Jose Maria Bermudo Mera, and Jan-Pieter D'Anvers. 2020. Supporting documentation: SABER: Mod-LWR based KEM (Round 3 Submission). *NIST Post-Quantum Cryptography Standardization Process* (2020).

[15] Mihir Bellare and Phillip Rogaway. 1993. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *CCS '93*. 62–73.

[16] Daniel J. Bernstein. 2001. Multidigit multiplication for mathematicians. http://cr.yp.to/papers.html#m3. (2001).

[17] Daniel J. Bernstein. 2016. S-unit attacks. (Aug. 2016). https://groups.google.com/g/cryptanalytic-algorithms/c/mCMdsFemzQk/m/3cewE8Q5BwAJ

[18] Daniel J. Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, and Christine van Vredendaal. 2017. NTRU Prime: Reducing Attack Surface at Low Cost. In *SAC 2017*, Vol. 10719. 235–260.

[19] Daniel J. Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, and Christine van Vredendaal. 2020. NTRU Prime: round 3. *NIST Post-Quantum Cryptography Standardization Process* (2020).

[20] Daniel J. Bernstein and Tanja Lange. 2021. Non-randomness of S-unit lattices. *IACR Cryptol. ePrint Arch.* (2021), 1428.

[21] Jean-François Biasse and Fang Song. 2016. Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields. In *SODA 2016*. 893–902.

[22] Nina Bindel, Mike Hamburg, Kathrin Hövelmanns, Andreas Hülsing, and Edoardo Persichetti. 2019. Tighter Proofs of CCA Security in the Quantum Random Oracle Model. In *TCC 2019*, Vol. 11892. 61–90.

[23] Avrim Blum, Adam Kalai, and Hal Wasserman. 2003. Noise-tolerant learning, the parity problem, and the statistical query model. *J. ACM* 50, 4 (2003), 506–519.

[24] Andrej Bogdanov, Siyao Guo, Daniel Masny, Silas Richelson, and Alon Rosen. 2016. On the Hardness of Learning with Rounding over Small Modulus. In *TCC 2016*, Vol. 9562. 209–224.

[25] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. 2011. Random Oracles in a Quantum World. In *ASIACRYPT 2011*, Vol. 7073. 41–69.

[26] Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. 2018. CRYSTALS - Kyber: A CCA-Secure Module-Lattice-Based KEM. In *IEEE EuroS&P 2018*. 353–367.

[27] Johannes Buchmann, Florian Göpfert, Rachel Player, and Thomas Wunderer. 2016. On the Hardness of LWE with Binary Error: Revisiting the Hybrid Lattice-Reduction and Meet-in-the-Middle Attack. In *AFRICACRYPT 2016*, Vol. 9646. 24–43.

[28] Peter Campbell, Michael Groves, and Dan Shepherd. 2014. Soliloquy: A cautionary tale. In *ETSI 2nd Quantum-Safe Crypto Workshop*, Vol. 3. 1–9.

[29] Cong Chen, Oussama Danba, Jeffrey Hoffstein, and Andreas Hulsing. 2020. NTRU submission. *NIST Post-Quantum Cryptography Standardization Process* (2020).

[30] Yuanmi Chen and Phong Q. Nguyen. 2011. BKZ 2.0: Better Lattice Security Estimates. In *ASIACRYPT 2011*, Vol. 7073. 1–20.

[31] Jung Hee Cheon, Minki Hhan, Seungwan Hong, and Yongha Son. 2019. A Hybrid of Dual and Meet-in-the-Middle Attack on Sparse and Ternary Secret LWE. *IEEE Access* 7 (2019), 89497–89506.

[32] Henri Cohen. 2012. *Advanced topics in computational number theory.* Vol. 193. Springer Science & Business Media.

[33] John H. Conway and Neil J. A. Sloane. 1982. Fast quantizing and decoding and algorithms for lattice quantizers and codes. *IEEE Trans. Inf. Theory* 28, 2 (1982), 227–231.

[34] John Horton Conway and Neil James Alexander Sloane. 2013. *Sphere packings, lattices and groups.* Vol. 290. Springer Science & Business Media.

[35] James W Cooley and John W Tukey. 1965. An algorithm for the machine calculation of complex Fourier series. *Mathematics of computation* 19, 90 (1965), 297–301.

[36] Don Coppersmith and Adi Shamir. 1997. Lattice Attacks on NTRU. In *EUROCRYPT '97*, Vol. 1233. 52–61.

[37] Jean-Sébastien Coron, Helena Handschuh, Marc Joye, Pascal Paillier, David Pointcheval, and Christophe Tymen. 2002. GEM: A Generic Chosen-Ciphertext Secure Encryption Method. In *CT-RSA 2002*, Vol. 2271. 263–276.

[38] Ronald Cramer, Léo Ducas, Chris Peikert, and Oded Regev. 2016. Recovering Short Generators of Principal Ideals in Cyclotomic Rings. In *EUROCRYPT 2016*, Vol. 9666. 559–585.

[39] Ronald Cramer, Léo Ducas, and Benjamin Wesolowski. 2017. Short Stickelberger Class Relations and Application to Ideal-SVP. In *EUROCRYPT 2017*, Vol. 10210. 324–348.

[40] Dana Dachman-Soled, Léo Ducas, Huijing Gong, and Mélissa Rossi. 2020. LWE with Side Information: Attacks and Concrete Security Estimation. In *CRYPTO 2020*, Vol. 12171. 329–358.

[41] Alexander W. Dent. 2003. A Designer's Guide to KEMs. In *IMACC 2003*, Vol. 2898. 133–151.

[42] Jintai Ding. 2013. Cryptographic system using pairing with errors. (April 11 2013). US Patent 9,246,675.

[43] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. 2021. Online-Extractability in the Quantum Random-Oracle Model. *IACR Cryptol. ePrint Arch.* (2021), 280.

[44] Alexandre Duc, Florian Tramèr, and Serge Vaudenay. 2015. Better Algorithms for LWE and LWR. In *EUROCRYPT 2015*, Vol. 9056. 173–202.

[45] Léo Ducas. 2018. Shortest Vector from Lattice Sieving: A Few Dimensions for Free. In *EUROCRYPT 2018*, Vol. 10820. 125–145.

[46] Léo Ducas, Vadim Lyubashevsky, and Thomas Prest. 2014. Efficient Identity-Based Encryption over NTRU Lattices. In *ASIACRYPT 2014*, Vol. 8874. 22–41.

[47] Julien Duman, Kathrin Hövelmanns, Eike Kiltz, Vadim Lyubashevsky, and Gregor Seiler. 2021. Faster Lattice-Based KEMs via a Generic Fujisaki-Okamoto Transform Using Prefix Hashing. 2722–2737.

[48] Julien Duman, Kathrin Hövelmanns, Eike Kiltz, Vadim Lyubashevsky, Gregor Seiler, and Dominique Unruh. 2021. A Thorough Treatment of Highly-Efficient NTRU Instantiations. *IACR Cryptol. ePrint Arch.* (2021), 1352.

[49] Kirsten Eisenträger, Sean Hallgren, Alexei Y. Kitaev, and Fang Song. 2014. A quantum algorithm for computing the unit group of an arbitrary degree number field. In *STOC 2014*. 293–302.

[50] Thomas Espitau, Antoine Joux, and Natalia Kharchenko. 2020. On a hybrid approach to solve small secret LWE. *Cryptology ePrint Archive* (2020).

[51] Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, and Vadim Lyubashevsky. 2020. Falcon: Fast-Fourier Lattice-based Compact Signatures over NTRU. *NIST Post-Quantum Cryptography Standardization Process* (2020).

[52] Pierre-Alain Fouque, Paul Kirchner, Thomas Pornin, and Yang Yu. 2022. BAT: Small and Fast KEM over NTRU Lattices. *IACR TCHES* 2022, 2 (2022), 240–265.

[53] Eiichiro Fujisaki and Tatsuaki Okamoto. 1999. Secure Integration of Asymmetric and Symmetric Encryption Schemes. In *CRYPTO' 99*, Vol. 1666. 537–554.

[54] Philippe Gaborit and Carlos Aguilar Melchor. 2011. Cryptographic method for communicating confidential information. (Feb. 17 2011). US Patent 9,094,189.

[55] Sanjam Garg, Craig Gentry, and Shai Halevi. 2013. Candidate Multilinear Maps from Ideal Lattices. In *EUROCRYPT 2013*, Vol. 7881. 1–17.

[56] W. Morven Gentleman and G. Sande. 1966. Fast Fourier Transforms: for fun and profit. In *AFIPS '66 (AFIPS Conference Proceedings)*, Vol. 29. 563–578.

[57] Werner H Greub. 2012. *Linear algebra*. Vol. 23. Springer Science & Business Media.

[58] Qian Guo and Thomas Johansson. 2021. Faster Dual Lattice Attacks for Solving LWE with Applications to CRYSTALS. In *ASIACRYPT 2021*, Vol. 13093. 33–62.

[59] Qian Guo, Thomas Johansson, and Paul Stankovski. 2015. Coded-BKW: Solving LWE Using Lattice Codes. In *CRYPTO 2015*, Vol. 9215. 23–42.

[60] Sean Hallgren. 2005. Fast quantum algorithms for computing the unit group and class group of a number field. In *STOC 2005*. 468–474.

[61] Philip S. Hirschhorn, Jeffrey Hoffstein, Nick Howgrave-Graham, and William Whyte. 2009. Choosing NTRUEncrypt Parameters in Light of Combined Lattice Reduction and MITM Approaches. In *ACNS 2009*,, Vol. 5536. 437–455.

[62] Jeffrey Hoffstein. 1996. NTRU: a new high speed public key cryptosystem. *presented at the rump session of Crypto 96* (1996).

[63] Jeffrey Hoffstein, Nick Howgrave-Graham, Jill Pipher, Joseph H. Silverman, and William Whyte. 2003. NTRUSIGN: Digital Signatures Using the NTRU Lattice. In *CT-RSA 2003*, Vol. 2612. 122–140.

[64] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. 1998. NTRU: A Ring-Based Public Key Cryptosystem. In *ANTS*, Vol. 1423. 267–288.

[65] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. 2017. A Modular Analysis of the Fujisaki-Okamoto Transformation. In *TCC 2017*, Vol. 10677. 341–371.

[66] Kathrin Hövelmanns, Eike Kiltz, Sven Schäge, and Dominique Unruh. 2020. Generic Authenticated Key Exchange in the Quantum Random Oracle Model. In *PKC 2020*, Vol. 12111. 389–422.

[67] Nick Howgrave-Graham. 2007. A Hybrid Lattice-Reduction and Meet-in-the-Middle Attack Against NTRU. In *CRYPTO 2007*, Vol. 4622. 150–169.

[68] N Howgrave-Graham and A Menezes. 2007. A hybrid meet-in-the-middle and lattice reduction attack on NTRU. In *CRYPTO*. 150–169.

[69] Andreas Hülsing, Joost Rijneveld, John M. Schanck, and Peter Schwabe. 2017. High-Speed Key Encapsulation from NTRU. In *CHES 2017*, Vol. 10529. 232–252.

[70] Katherine Jarvis and Monica Nevins. 2015. ETRU: NTRU over the Eisenstein integers. *Des. Codes Cryptogr.* 74, 1 (2015), 219–242.

[71] Haodong Jiang, Zhenfeng Zhang, Long Chen, Hong Wang, and Zhi Ma. 2018. IND-CCA-Secure Key Encapsulation Mechanism in the Quantum Random Oracle Model, Revisited. In *CRYPTO 2018*, Vol. 10993. 96–125.

[72] Haodong Jiang, Zhenfeng Zhang, and Zhi Ma. 2019. Key Encapsulation Mechanism with Explicit Rejection in the Quantum Random Oracle Model. In *PKC 2019*, Vol. 11443. 618–645.

[73] Haodong Jiang, Zhenfeng Zhang, and Zhi Ma. 2019. Tighter Security Proofs for Generic Key Encapsulation Mechanism in the Quantum Random Oracle Model. In *PQCrypto 2019*, Vol. 11505. 227–248.

[74] Haodong Jiang, Zhenfeng Zhang, and Zhi Ma. 2021. On the Non-tightness of Measurement-Based Reductions for Key Encapsulation Mechanism in the Quantum Random Oracle Model. In *ASIACRYPT 2021*, Vol. 13090. 487–517.

[75] Ravi Kannan. 1987. Minkowski's Convex Body Theorem and Integer Programming. *Math. Oper. Res.* 12, 3 (1987), 415–440.

[76] Paul Kirchner and Pierre-Alain Fouque. 2015. An Improved BKW Algorithm for LWE with Applications to Cryptography and Lattices. In *CRYPTO 2015*, Vol. 9215. 43–62.

[77] Paul Kirchner and Pierre-Alain Fouque. 2017. Revisiting Lattice Attacks on Overstretched NTRU Parameters. In *EUROCRYPT 2017*, Vol. 10210. 3–26.

[78] Veronika Kuchta, Amin Sakzad, Damien Stehlé, Ron Steinfeld, and Shifeng Sun. 2020. Measure-Rewind-Measure: Tighter Quantum Random Oracle Model Proofs for One-Way to Hiding and CCA Security. In *EUROCRYPT 2020*, Vol. 12107. 703–728.

[79] Adeline Langlois and Damien Stehlé. 2015. Worst-case to average-case reductions for module lattices. *Des. Codes Cryptogr.* 75, 3 (2015), 565–599.

[80] Adeline Langlois, Damien Stehlé, and Ron Steinfeld. 2014. GGHLite: More Efficient Multilinear Maps from Ideal Lattices. In *EUROCRYPT 2014*, Vol. 8441. 239–256.

[81] Richard Lindner and Chris Peikert. 2011. Better Key Sizes (and Attacks) for LWE-Based Encryption. In *CT-RSA 2011*, Vol. 6558. 319–339.

[82] Mingjie Liu and Phong Q. Nguyen. 2013. Solving BDD by Enumeration: An Update. In *CT-RSA 2013*, Vol. 7779. 293–309.

[83] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. 2012. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In *STOC 2012*. 1219–1234.

[84] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. 2010. On Ideal Lattices and Learning with Errors over Rings. In *EUROCRYPT 2010*, Vol. 6110. 1–23.

[85] Vadim Lyubashevsky and Gregor Seiler. 2019. NTTRU: Truly Fast NTRU Using NTT. *IACR TCHES* 2019, 3 (2019), 180–201.

[86] MATZOV. 2022. Report on the Security of LWE: Improved Dual Lattice Attack. (April 2022). https://doi.org/10.5281/zenodo.6412487

[87] Daniele Micciancio and Oded Regev. 2008. Post-Quantum Cryptography, chapter Lattice-based Cryptography. *Computing* 85, 1-2 (2008), 105–125.

[88] Robert T. Moenck. 1976. Practical fast polynomial multiplication. In *SYMSAC 1976*. ACM, 136–148.

[89] Phong Nguyen. 2021. Boosting the hybrid attack on NTRU: torus LSH, permuted HNF and boxed sphere. In *NIST Third PQC Standardization Conference*.

[90] NIST. 2016. Post-Quantum Cryptography, Round 1 Submissions. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/round-1-submissions. (2016).

[91] NIST. 2019. Post-Quantum Cryptography, Round 2 Submissions. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/round-2-submissions. (2019).

[92] NIST. 2020. Post-Quantum Cryptography, Round 3 Submissions. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/round-3-submissions. (2020).

[93] Tatsuaki Okamoto and David Pointcheval. 2001. REACT: Rapid enhanced-security asymmetric cryptosystem transform. In *CT-RSA 2001*. 159–174.

[94] Alice Pellet-Mary, Guillaume Hanrot, and Damien Stehlé. 2019. Approx-SVP in Ideal Lattices with Pre-processing. In *EUROCRYPT 2019*, Vol. 11477. 685–716.

[95] John M Pollard. 1971. The fast Fourier transform in a finite field. *Mathematics of computation* 25, 114 (1971), 365–374.

[96] Prasanna Ravi, Martianus Frederic Ezerman, Shivam Bhasin, Anupam Chattopadhyay, and Sujoy Sinha Roy. 2022. Will You Cross the Threshold for Me? Generic Side-Channel Assisted Chosen-Ciphertext Attacks on NTRU-based KEMs. *IACR TCHES* 2022, 1 (2022), 722–761.

[97] Oded Regev. 2009. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM* 56, 6 (2009), 34:1–34:40.

[98] Tsunekazu Saito, Keita Xagawa, and Takashi Yamakawa. 2018. Tightly-Secure Key-Encapsulation Mechanism in the Quantum Random Oracle Model. In *EUROCRYPT 2018*, Vol. 10822. 520–551.

[99] Claus-Peter Schnorr and M. Euchner. 1994. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Math. Program.* 66 (1994), 181–199.

[100] Yongha Son and Jung Hee Cheon. 2019. Revisiting the Hybrid attack on sparse and ternary secret LWE. *IACR Cryptol. ePrint Arch.* (2019), 1019.

[101] Damien Stehlé and Ron Steinfeld. 2011. Making NTRU as Secure as Worst-Case Problems over Ideal Lattices. In *EUROCRYPT 2011*, Vol. 6632. 27–47.

[102] Ron Steinfeld. 2014. NTRU cryptosystem: Recent developments and emerging mathematical problems in finite polynomial rings. *Algebraic Curves and Finite Fields* (2014), 179–212.

[103] Maryna Viazovska. 2017. The sphere packing problem in dimension 8. *Annals of Mathematics* (2017), 991–1015.

[104] Yang Wang and Mingqiang Wang. 2018. Provably Secure NTRUEncrypt over Any Cyclotomic Field. In *SAC 2018*, Vol. 11349. 391–417.

[105] Lawrence C. Washington. 1997. *Introduction to cyclotomic fields*. Graduate Texts in Mathematics 83, Springer-Verlag.

[106] Thomas Wunderer. 2019. A detailed analysis of the hybrid lattice-reduction and meet-in-the-middle attack. *J. Math. Cryptol.* 13, 1 (2019), 1–26.

[107] Yang Yu, Guangwu Xu, and Xiaoyun Wang. 2017. Provably Secure NTRU Instances over Prime Cyclotomic Rings. In *PKC 2017*, Vol. 10174. 409–434.

[108] Yang Yu, Guangwu Xu, and Xiaoyun Wang. 2017. Provably Secure NTRUEncrypt over More General Cyclotomic Rings. *IACR Cryptol. ePrint Arch.* (2017), 304.

A ON CCA SECURITY REDUCTION OF KEM IN THE ROM AND THE QROM

Generic constructions of an efficient IND-CCA secure KEM are well studied in [41, 65], which are essentially various KEM variants of Fujisaki-Okamoto (FO) transformation [53] and GEM/REACT transformation [37, 93]. The work [65] gives a modular analysis of various FO transformations in the ROM and the QROM, and summarizes some practical FO transformations that are widely used to construct an IND-CCA secure KEM from a passive secure PKE (e.g., OW-CPA and IND-CPA), including the following transformations $\mathrm{FO}^{\perp}$, $\mathrm{FO}_m^{\perp}$, $\mathrm{FO}^{\not\perp}$, $\mathrm{FO}_m^{\not\perp}$, $\mathrm{U}^{\not\perp}$ and $\mathrm{U}_m^{\not\perp}$, etc, where m (without m) means $K = H(m)$ $(K = H(m, c))$, $\not\perp$ $(\perp)$ means implicit (explicit) rejection.

$\mathrm{FO}^{\perp}$, $\mathrm{FO}_m^{\perp}$, $\mathrm{FO}^{\not\perp}$ and $\mathrm{FO}_m^{\not\perp}$ are the most common transformations used in NIST PQC. According to [65], in the ROM, the reduction bound of these four transformations are all $\epsilon' \leq \epsilon_{CPA} + q'\delta$ and $\epsilon' \leq q'\epsilon_{OW} + q'\delta$, where ϵ' is the advantage of an adversary against IND-CCA security of KEM, ϵ_{CPA} (ϵ_{OW}) is the the advantage of an adversary against IND-CPA (OW-CPA) security of the underlying PKE, q' is the total number of hash queries, and δ is the error probability. Notice that in order to keep the comparison lucid, we ignore the small constant factors and additional inherent summands. The reduction is tight for IND-CPA secure PKE, but it has a loss factor q' for OW-CPA secure PKE in the ROM. However, all of their reduction bounds in the QROM suffer from a quartic loss, i.e., $\epsilon' \leq q'\sqrt{q'\sqrt{\epsilon_{OW}} + q'^2\delta}$ with an additional hash in [65]. Later, the bound of $\mathrm{FO}^{\not\perp}$ is improved as follows: $\epsilon' \leq q'\sqrt{\epsilon_{OW}} + q'\sqrt{\delta}$ without additional hash in [71], $\epsilon' \leq \sqrt{q'\epsilon_{CPA}} + q'\sqrt{\delta}$ with semi-classical oracles [8] in [73], $\epsilon' \leq \sqrt{q'\epsilon_{CPA}} + q'^2\delta$ with double-sided OW2H lemma in [22], and $\epsilon' \leq q'^2\epsilon_{CPA} + q'^2\delta$ with measure-rewind-measure technique in [78]. The bound of $\mathrm{FO}_m^{\not\perp}$ is improved as follows: $\epsilon' \leq q'\sqrt{\epsilon_{OW}} + q'\sqrt{\delta}$ without additional hash in [71], $\epsilon' \leq \sqrt{q'\epsilon_{CPA}} + q'^2\delta$ with disjoint simulatability in [66], $\epsilon' \leq \sqrt{q'\epsilon_{CPA}} + q'^2\delta$ with prefix hashing in [47]. The bound of $\mathrm{FO}_m^{\perp}$ is improved as follows: $\epsilon' \leq q'\sqrt{\epsilon_{OW}} + q'\sqrt{\delta}$ and $\epsilon' \leq \sqrt{q'\epsilon_{CPA}} + q'\sqrt{\delta}$ with extra hash in [72], $\epsilon' \leq q'\sqrt{\epsilon_{OW}} + q'^2\sqrt{\delta}$ without extra hash in [43].

There also exists some transformations with tight reduction for deterministic PKE (DPKE) with disjoint simulatability and perfect correctness, for example, a variant of $\mathrm{U}_m^{\not\perp}$ proposed in [98]. In the case that the underlying PKE is non-deterministic, all known bounds are of the form $O(\sqrt{q'\epsilon_{CPA}})$ and $O(q'\sqrt{\epsilon_{OW}})$ as we introduce above, with the exception of [78]. The work [74] shows that the measurement-based reduction involving no rewinding will inevitably incur a quadratic loss of the security in the QROM. In another word, as for the underlying PKE, the IND-CPA secure PKE has a tighter reduction bound than the OW-CPA secure PKE. It also significantly leads us to construct an IND-CPA secure PKE for tighter reduction bound of the resulting IND-CCA secure KEM.

Some discussions are presented here for comparing the reduction bounds of CTRU and other NTRU-based KEM schemes. Most of the existing NTRU-based encryption schemes can only achieve OW-CPA security. NTRU-HRSS and SNTRU Prime construct the KEM schemes from OW-CPA DPKEs via $\mathrm{U}_m^{\not\perp}$ variants. Although they can reach tight CCA reductions with extra assumptions in the (Q)ROM [19, 29], there is a disadvantage that some extra computation is needed to recover the randomness in the decryption algorithms.

Determinism is a much stricter condition, thus some NTRU-based PKEs prefer to be non-deterministic (i.e., randomized). NTTRU applies $\mathrm{FO}_m^{\perp}$ to build an IND-CCA KEM from an OW-CPA randomized PKE [85]. According to [43, 65], its IND-CCA reduction bounds are not-tight in both the ROM $(O(q'\epsilon_{OW}))$ and the QROM $(O(q'\sqrt{\epsilon_{OW}}))$.

NTRU-C is the general form of NTRU-C_{3457}^{768}. NTRU-C uses a slightly different way that it first constructs an IND-CPA PKE from an OW-CPA NTRU-based PKE via ACWC_0 transformation [48], and then transforms it into an IND-CCA KEM via $\mathrm{FO}_m^{\perp}$. Note that ACWC_0 brings two terms of ciphertexts, where the extra term of ciphertexts costs 32 bytes. The IND-CPA security of the resulting after-ACWC_0 PKE can be tightly reduced to the OW-CPA security of the underlying before-ACWC_0 PKE in the ROM. However, there is a quadratic loss advantage in the QROM, i.e., $\epsilon_{CPA} \leq q'\sqrt{\epsilon_{OW}}$. In the ROM, the advantage of the adversary against IND-CCA security of KEM is tightly reduced to that of the adversary against IND-CPA security of after-ACWC_0 PKE, and consequently is tightly reduced to that of the adversary against OW-CPA security of before-ACWC_0 PKE. However, in the QROM, there is no known direct reduction proof about $\mathrm{FO}_m^{\perp}$ from IND-CPA PKE to IND-CCA KEM without additional hash. The reduction bound of $\mathrm{FO}_m^{\perp}$ in the QROM in [43] only aims at the underlying OW-CPA PKE. Since the IND-CPA security implies OW-CPA security [65], the reduction bound of IND-CCA KEM to before-ACWC_0 OW-CPA PKE will suffer from the quartic advantage loss in the QROM. That is, if the adversary has ϵ_{OW} advantage against the before-ACWC_0 OW-CPA PKE, then it has $O(q'^{1.5}\sqrt[4]{\epsilon_{OW}})$ advantage against the resulting IND-CCA KEM in the QROM. On the other hand, with an additional hash, a better bound of $\mathrm{FO}_m^{\perp}$ for after-ACWC_0 IND-CPA PKE can be achieved, i.e., $O(\sqrt{q'\epsilon_{CPA}})$ advantage against the resulting IND-CCA KEM in the QROM [72] at the cost of some extra ciphertext burden. ACWC_0 also has an effect on the efficiency, since an extra transformation from OW-CPA PKE to IND-CPA PKE is also relatively time-consuming.

Our CTRU seems to be more simple, compact, efficient and memory-saving than other NTRU-based KEM schemes, along with a tight bound in the ROM and a tighter bound in the QROM for IND-CCA security. When compared to NTRU-HRSS, SNTRU Prime and NTRU-C, an obvious efficiency improvement of our CTRU is due to the fact that there is no extra requirement of recovering randomness in decryption algorithm or reinforced transformation to obtain IND-CPA security. CTRU.PKE can achieve IND-CPA security in the case that its ciphertext can be only represented by a single polynomial, without any extra ciphertext term like NTRU-C. Starting from our IND-CPA PKE to construct KEM with $\mathrm{FO}_{ID(pk),m}^{\not\perp}$, the reduction bound of IND-CCA security is tightly reduced to IND-CPA security in the ROM $(\epsilon' \leq O(\epsilon_{CPA})$, restated), so it is tightly reduced to the underlying hardness assumptions. We also have the known best bound in the QROM $(\epsilon' \leq O(\sqrt{q'\epsilon_{CPA}})$, restated) according to [47], which is better than those in NTTRU and NTRU-C.

A.1 CCA Security in Multi-User Setting

We remark that, the work [47] originally gives the multi-user/challenge IND-CCA reduction bound of $\mathrm{FO}^{\not\perp}_{ID(pk),m}$ in the ROM and the QROM. We adapt the results from Theorem 3.1 and Theorem 3.2 in [47] into the single-user/challenge setting of CTRU, which is only for ease of fair comparisons as other KEM schemes only utilize single-user/challenge FO transformations. As CTRU.PKE is IND-CPA secure, another advantage of using $\mathrm{FO}^{\not\perp}_{ID(pk),m}$ is that CTRU can be improved to enjoy the multi-user/challenge IND-CCA security as well. To address this issue, some adjustments are needed as follows. Unlike the single-user/challenge setting, the adversary (against the n'-user/q_C-challenge IND-CPA security of the underlying PKE) is given the public keys of n' users, and is allowed to make at most q_C challenge queries w.r.t. the same challenge plaintext m_b chosen by the challenger. According to [47], based on the single-user/challenge IND-CPA security of the underlying PKE, the formal multi-user/challenge IND-CCA security of the resulting KEM is given in Theorem A.1.

Theorem A.1 (n'-user/q_C-challenge IND-CCA security in the ROM and the QROM [47]). *Following [47], we will use (or recall) the following terms in the concrete security statements.*

- *n'-user error probability $\delta(n')$ [47].*
- *Min-entropy ℓ [53] of $ID(pk)$, i.e., $\ell = H_\infty(ID(pk))$, where $(pk, sk) \leftarrow CTRU.PKE.KeyGen$.*
- *Bit-length ι of the secret seed $z \in \{0,1\}^\iota$.*
- *Maximal number of (Q)RO queries q_H.*
- *Maximal number of decapsulation queries q_D.*
- *Maximal number of challenge queries q_C.*

For any (quantum) adversary A *against the (n', q_C)-IND-CCA security of CTRU.KEM, there exits a (quantum) adversary* B *against the (n', q_C)-IND-CPA security of CTRU.PKE with roughly the same running time of* A*, such that:*

- *In the ROM, it holds that* $\boldsymbol{Adv}^{(n',q_C)\text{-}IND\text{-}CCA}_{CTRU.KEM}(\mathsf{A}) \leq$

$$2\left(\boldsymbol{Adv}^{(n',q_C)\text{-}IND\text{-}CPA}_{CTRU.PKE}(\mathsf{B}) + \frac{(q_H+q_C)q_C}{|\mathcal{M}|}\right)+\frac{q_H}{2^\iota}+(q_H+q_D)\delta(n')+\frac{n'^2}{2^\ell};$$

- *In the QROM, it holds that* $\boldsymbol{Adv}^{(n',q_C)\text{-}IND\text{-}CCA}_{CTRU.KEM}(\mathsf{A}) \leq$

$$2\sqrt{q_{HD}\boldsymbol{Adv}^{(n',q_C)\text{-}IND\text{-}CPA}_{CTRU.PKE}(\mathsf{B})} + 4q_{HD}\sqrt{\frac{q_C \cdot n'}{|\mathcal{M}|}} + 4(q_H+1)\sqrt{\frac{n'}{2^\iota}}$$

$$+16q_{HD}^2\delta(n') + \frac{q_C^2}{|\mathcal{M}|} + \frac{n'^2}{2^\ell}, \textit{ where } q_{HD} := q_H + q_D + 1.$$

B THE FORM OF POLYNOMIAL PRODUCT IN $\mathbb{Z}[X]/(X^N - X^{N/2} + 1)$

Following the methodology in [85], the general form of the polynomial product of $f = \sum_{i=0}^{n-1} f_i x^i$ and $g = \sum_{i=0}^{n-1} g_i x^i$ in the ring $\mathbb{Z}[x]/(x^n - x^{n/2} + 1)$ is presented via a matrix-vector multiplication

$$h = \begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_{n-1} \end{bmatrix} = \begin{bmatrix} \mathbf{L}-\mathbf{U} & -\mathbf{F}-\mathbf{U} \\ \mathbf{F}+\mathbf{U} & \mathbf{F}+\mathbf{L} \end{bmatrix} \cdot \begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_{n-1} \end{bmatrix}, \qquad (6)$$

where $\mathbf{F}, \mathbf{L}, \mathbf{U}$ are the $n/2$-dimension Toeplitz matrices as follows:

$$\mathbf{F} = \begin{bmatrix} f_{n/2} & f_{n/2-1} & \cdots & f_1 \\ f_{n/2+1} & f_{n/2} & \cdots & f_2 \\ \vdots & \vdots & \ddots & \vdots \\ f_{n-1} & f_{n-2} & \cdots & f_{n/2} \end{bmatrix}, \ \mathbf{L} = \begin{bmatrix} f_0 & 0 & \cdots & 0 \\ f_1 & f_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ f_{n/2-1} & f_{n/2-2} & \cdots & f_0 \end{bmatrix},$$

$$\mathbf{U} = \begin{bmatrix} 0 & f_{n-1} & \cdots & f_{n/2+1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & f_{n-1} \\ 0 & 0 & \cdots & 0 \end{bmatrix}.$$

The computation about the error probability in Theorem 4.2 is based on the polynomial product in $\mathbb{Z}[x]/(x^n - x^{n/2} + 1)$. The whole product is divided into two parts through the form of partitioned matrices. In the formula (6), each coefficient of the bottom half of the product, i.e., $[\mathbf{F}+\mathbf{U}\ \mathbf{F}+\mathbf{L}] \cdot [g_0, g_1, \ldots, g_{n-1}]^T$, is obtained from the sum of $n/2$ terms of the form $\xi_{i,i',j,j'} = f_i g_j + (f_i + f_{i'})g_{j'}$. Similarly, the coefficient of the l-th row of the upper half, i.e., $[\mathbf{L}-\mathbf{U}\ -\mathbf{F}-\mathbf{U}] \cdot [g_0, g_1, \ldots, g_{n-1}]^T$, is the sum of $n/2 - l$ terms of the form $\xi_{i,i',j,j'}$, and l terms of the form $\vartheta_{i,i',j,j'} = f_i g_j + f_{i'} g_{j'}$. As suggested in [85], the distribution of $\xi_{i,i',j,j'}$ has a "wider" probability distribution than that of $\vartheta_{i,i',j,j'}$. To bound the error probability correctly, we should consider the widest distribution consisting of sums of $n/2$ random variables with the distribution of $\xi_{i,i',j,j'}$. Therefore, in our correctness analysis, the term $gr + ef + (\varepsilon_1 + \frac{q}{q_2}\varepsilon_2)f$ in Theorem 4.2 will be computed from this methodology.

C S-UNIT ATTACK

Here we refer to [20] to briefly introduce S-unit attack.

S-unit attack begins with a nonzero $v \in I$ and outputs v/u, but now u is allowed to range over a larger subset of K^* , specifically the group of S-units.

Here S is a finite set of places, a subset of the set V mentioned above. There are two types of places:

- The "infinite places" are labeled $1, 3, 5, \ldots, n-1$, except that for $n = 1$ there is one infinite place labeled 1. The entry at place j in $\log \alpha$ is defined as $2 \log |\sigma_j(\alpha)|$, except that the factor 2 is omitted for $n = 1$. The set of all infinite places is denoted ∞, and is required to be a subset of S.
- For each nonzero prime ideal P of R, there is a "finite place" which is labeled as P. The entry at place P in $\log \alpha$ is defined as $-(ord_P \alpha) \log |(R/P)|$, where $ord_P \alpha$ is the exponent of P in the factorization of α as the product of powers of prime ideals. There are many choices of S here. It focuses on the following form of S: choose a parameter y, and take $P \in S$ if and only if $|(R/P)| \leq y$.

The group U_S of S-units of K is, by definition, the set of elements $u \in K^*$such that the vector $\log u$ is supported on S, i.e., it is 0 at every place outside S. The S-unit lattice is the lattice $\log U_S$, which has rank $|S - 1|$.

Short v/u again corresponds to short $\log v - \log u$, but it is required to ensure that $v/u \in I$, i.e., $ord_P(v/u) \geq ord_P I$ for each finite place P. This was automatic for unit attacks but is not automatic for general S-unit attacks. One thus wants to find a vector $\log u$ in the S-unit lattice $\log U_S$ that is close to $\log v$ in the following sense:

Table 5: Comparison between CTRU and Saber.

Schemes	Assumptions	Reduction	Rings	n	q	$\lvert pk\rvert$	$\lvert ct\rvert$	B.W.	(Sec.C, Sec.Q)	δ
CTRU (Ours)	NTRU, RLWE	IND-CPA RPKE	$\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$	512	3457	768	640	1408	(118,107)	2^{-144}
				768	3457	1152	960	2112	(181,164)	2^{-187}
				1024	3457	1536	1408	2944	(255,231)	2^{-206}
Saber [14]	MLWR	IND-CPA RPKE	$\mathbb{Z}_q[x]/(x^{n/k} + 1)$ $k = 2, 3, 4$	512	8192	672	736	1408	(118,107)	2^{-120}
				768	8192	992	1088	2080	(189,172)	2^{-136}
				1024	8192	1312	1472	2784	(260,236)	2^{-165}

$\log u$ is close to $\log v$ at the infinite places, and $ord_P u$ is close to but no greater than $ord_P v - ord_P I$.

As for closeness, as a preliminary step, if $ord_P v < ord_P I$ for some P, update v by multiplying it by a generator of $P\hat{P}$ (or, if possible, of P) as explained above, and repeat this step. Then $v \in I$. Next, if some u in the list has v/u shorter than v and $v/u \in I$, replace v with v/u, and repeat this step. Output the final v.

As an extreme case, if $S = \infty$ (the smallest possible choice, not including any P), then $U_S = R^*$: the S-units of K are the units of R, the S-unit lattice is the unit lattice, and S-unit attacks are the same as unit attacks. Extending S to include more and more prime ideals P gives S-unit attacks the ability to modify more and more places in $\log v$.

D COMPARISONS BETWEEN CTRU AND SABER

Saber [14] is based on the MLWR assumption that can be viewed as *derandomized* MLWE [7, 13]. The reduction from (M)LWR to (M)LWE is not tight, which suffers from a polynomial loss in the case of small modulus [24]. In addition, the security estimation made by Saber [14] is slightly different from those of other schemes. Therefore, for ease of fair comparisons, we do not make a direct comparison between Saber and other lattice-based schemes. However, for the sake of completeness, the comparison between CTRU and Saber is still given in Table 5. From Table 5, at about the security levels, CTRU enjoys smaller ciphertext sizes and lower error probabilities. The sizes of public keys of Saber are relatively smaller due to the MLWR trick used.

We also implement Saber with the modification that its FO transformation has also been changed to $\mathrm{FO}^{\not\perp}_{ID(pk),m}$ as in [47]. The benchmark results are summarized in Table 6. It can be seen that the Encaps process of CTRU is still the most efficient. The efficiency improvements in the KeyGen and the Decaps of Saber are benefited from the fact: most arithmetic operations can be conducted by bitwise operations since all the moduli used in Saber are power-of-two.

Table 6: CPU cycles of CTRU and Saber (in kilo cycles).

Schemes	KeyGen	Encaps	Decaps
CTRU-768	8.2×10^3	80.9	151.8
Saber-768 [14]	94.7	109.7	138.9

E CNTR: CONSTRUCTION AND ANALYSIS

We propose a variant of CTRU, named CNTR, which is based on the NTRU assumption [64] and the RLWR assumption [13]. Here, CNTR stands for “Compact NTRu based on RLWR". CNTR is also usually the abbreviation of container, which has the meaning CNTR is an economically concise yet powerful key encapsulation mechanism.

E.1 Proposal Description

Our CNTR.PKE scheme is specified in Algorithm 12-14. Restate that $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$, where n and q are the ring parameters. Let q_2 be the modulus, which is usually set to be a power of two that is less than q. Let p be the message space modulus, satisfying $\gcd(q, p) = 1$. Let Ψ and Ψ_r be the distribution over $\mathcal{R}$. For presentation simplicity, the secret terms, f' and g are taken from Ψ, and r is taken from Ψ_r. Actually, Ψ and Ψ_r can be different distributions. Let $\mathcal{M} = \{0, 1\}^{n/2}$ denote the message space, where each $m \in \mathcal{M}$ can be seen as a $\frac{n}{2}$-dimension polynomial with coefficients in $\{0, 1\}$. The PolyEncode algorithm and PolyDecode algorithm are the same as Algorithm 7 and 8, respectively.

Algorithm 12 CNTR.PKE.KeyGen()

```
$f', g \leftarrow \Psi$
$f := pf' + 1$
If $f$ is not invertible in $\mathcal{R}_q$, restart.
$h := g/f$
return $(pk := h, sk := f)$
```

Algorithm 13 CNTR.PKE.Enc($pk = h$, $m \in \mathcal{M}$)

```
$r \leftarrow \Psi_r$
$\sigma := hr$
$c := \left\lfloor \frac{q_2}{q}(\sigma + \mathrm{PolyEncode}(m)) \right\rceil \bmod q_2$
return $c$
```

Algorithm 14 CNTR.PKE.Dec($sk = f$, c)

```
$m := \mathrm{PolyDecode}\left(cf \bmod {}^{\pm}q_2\right)$
return $m$
```

Unlike the encryption algorithm of CTRU (see Algorithm 5), that of CNTR has the following distinctions: (1) the error polynomial is moved; (2) the rounding of PolyEncode algorithm is moved.

Our CNTR.KEM scheme is constructed in the same way as CTRU.KEM, via the FO transformation $\mathrm{FO}^{\not\perp}_{ID(pk),m}$ [47]. The algorithms of CNTR.KEM can be referred to Algorithm 9-11.

E.2 Correctness Analysis

Theorem E.1. *Let Ψ and Ψ_r be the distribution over the ring $\mathcal{R}$, and q, q_2, p be positive integers. q_2 is an even number that is less than q. Let $f', g \leftarrow \Psi$ and $r \leftarrow \Psi_r$. Let $\varepsilon \leftarrow \chi$, where χ is the distribution over $\mathcal{R}$ defined as follows: Sample $u \overset{\$}{\leftarrow} [-\frac{q}{2q_2}, \frac{q}{2q_2}) \cap \mathbb{Z}$ and output $-\frac{q_2}{q}u$. Let Err_i be the i-th octet of $gr + \frac{q}{q_2}\varepsilon f$. Denote $1 - \delta = Pr\left[\|Err_i\|_{q,2} < \frac{q}{2}\right]$. Then, the error probability of CNTR is δ.*

Proof. The proof is similar to that of Theorem 4.2. The main observation is that the computation of the ciphertext c is equivalent to

$$\begin{aligned} c &= \left\lfloor \frac{q_2}{q}(\sigma + \mathrm{PolyEncode}(m)) \right\rceil \bmod q_2 \\ &= \left\lfloor \frac{q_2}{q}hr + \frac{q_2}{q} \cdot \frac{q}{2}s \right\rceil \bmod q_2 \\ &= \left\lfloor \frac{q_2}{q}hr \right\rceil + \frac{q_2}{2}s \bmod q_2 \\ &= \frac{q_2}{q}hr + \varepsilon + \frac{q_2}{2}s \bmod q_2 \end{aligned} \tag{7}$$

for even $q_2 < q$. The term $\lfloor \frac{q_2}{q}hr \rceil$ indicates an RLWR sample, and the term $\frac{q_2}{2}s$ implies the encoding output of m via the scalable E_8 lattice w.r.t. the scale factor $\frac{q_2}{2}$.

Similar to [14], we can roughly regard the distribution of $-\frac{q}{q_2}\varepsilon = hr - \frac{q}{q_2}\lfloor \frac{q_2}{q}hr \rceil$ as the uniform distribution over $[-\frac{q}{2q_2}, \frac{q}{2q_2}) \cap \mathbb{Z}$. Therefore, the distribution of ε can be subject to the distribution χ defined as in the theorem statement.

Similarly, we have

$$\begin{aligned} m &= \mathrm{PolyDecode}_{E_8''}\left(cf \bmod {}^{\pm}q_2\right) \\ &= \mathrm{PolyDecode}_{E_8'}\left((\frac{q}{q_2}c)f \bmod {}^{\pm}q \right), \end{aligned}$$

thereby $(\frac{q}{q_2}c)f = \frac{q}{2}s + gr + \frac{q}{q_2}\varepsilon f$. Each octet of $\frac{q}{2}s$ is essentially a lattice point in the scalable E_8 lattice w.r.t. the scale factor $\frac{q}{2}$, which we denoted by $\frac{q}{2}(\mathbf{k}_i\mathbf{H} \bmod 2)$. From Theorem 3.1 we know that to recover $\mathbf{k}_i$, it should hold $\|\mathrm{Err}_i\|_{q,2} < \frac{q}{2}$, where Err_i is the i-th octet of $gr + \frac{q}{q_2}\varepsilon f$.

□

E.3 Provable Security

Definition E.2 (RLWR assumption [13]). Let $q > p \geq 2$ be integers. Let Ψ_r be a distribution over a polynomial ring R. Let $R_q = R/qR$ and $R_p = R/pR$ be the quotient rings. The (decisional) Ring-Learning with rounding (RLWR) assumption is to distinguish uniform samples $(h, c) \overset{\$}{\leftarrow} R_q \times R_p$ from samples $(h, c) \in R_q \times R_p$ where $h \overset{\$}{\leftarrow} R_q$ and $c = \lfloor \frac{p}{q}hr \rceil \bmod p$ with $r \leftarrow \Psi_r$. It is hard if the advantage $\mathbf{Adv}^{\mathrm{RLWR}}_{R,\Psi_r}(\mathsf{A})$ of any probabilistic polynomial time adversary A is negligible, where $\mathbf{Adv}^{\mathrm{RLWR}}_{R,\Psi_r}(\mathsf{A}) =$

$$\left| \Pr\left[b' = 1 : \begin{array}{c} h \overset{\$}{\leftarrow} R_q; r \leftarrow \Psi_r; \\ c = \lfloor \frac{p}{q}hr \rceil \bmod p \in R_p; b' \leftarrow \mathsf{A}(h, c) \end{array} \right] - \Pr\left[b' = 1 : h \overset{\$}{\leftarrow} R_q; c \overset{\$}{\leftarrow} R_p; b' \leftarrow \mathsf{A}(h, c) \right] \right|.$$

Theorem E.3 (IND-CPA security of CNTR.PKE). *For any adversary* A, *there exits adversaries* B *and* C *such that $\boldsymbol{Adv}^{IND\text{-}CPA}_{CNTR.PKE}(\mathsf{A}) \leq \boldsymbol{Adv}^{NTRU}_{\mathcal{R}_q,\Psi}(\mathsf{B}) + \boldsymbol{Adv}^{RLWR}_{\mathcal{R},\Psi_r}(\mathsf{C})$.*

Proof. We complete our proof through a sequence of games $\mathbf{G}_0$, $\mathbf{G}_1$ and $\mathbf{G}_2$. Let A be the adversary against the IND-CPA security experiment. Denote by $\mathbf{Succ}_i$ the event that A wins in the game $\mathbf{G}_i$, that is, A outputs b' such that $b' = b$ in $\mathbf{G}_i$.

Game $\mathbf{G}_0$. This game is the original IND-CPA security experiment. Thus, $\mathbf{Adv}^{\mathrm{IND\text{-}CPA}}_{\mathrm{CNTR.PKE}}(\mathsf{A}) = |\Pr[\mathbf{Succ}_0] - 1/2|$.

Game $\mathbf{G}_1$. This game is the same as $\mathbf{G}_0$, except that replacing the public key $h = g/f$ in the KeyGen by $h \overset{\$}{\leftarrow} \mathcal{R}_q$. To distinguish $\mathbf{G}_1$ from $\mathbf{G}_0$ is equivalent to solve an NTRU problem. More precisely, there exits an adversary B with the same running time as that of A such that $|\Pr[\mathbf{Succ}_0] - \Pr[\mathbf{Succ}_1]| \leq \mathbf{Adv}^{\mathrm{NTRU}}_{\mathcal{R}_q,\Psi}(\mathsf{B})$.

Game $\mathbf{G}_2$. This game is the same as $\mathbf{G}_1$, except that using random elements from $\mathcal{R}_{q_2}$ to replace $\lfloor \frac{q_2}{q}hr \rceil$ of $c = \lfloor \frac{q_2}{q}hr \rceil + \frac{q_2}{2}s \bmod q_2$ (see the formula (7)) in the encryption where the term $\frac{q_2}{2}s$ implies the encoding output of the given challenge plaintext m_b via the scalable E_8 lattice w.r.t. the scale factor $\frac{q_2}{2}$. Similarly, there exits an adversary C with the same running time as that of A such that $|\Pr[\mathbf{Succ}_1] - \Pr[\mathbf{Succ}_2]| \leq \mathbf{Adv}^{\mathrm{RLWR}}_{\mathcal{R},\Psi_r}(\mathsf{C})$.

In Game $\mathbf{G}_2$, the information of the challenge plaintext m_b is perfectly hidden by the uniformly random element from $\mathcal{R}_{q_2}$. Hence, the advantage of the adversary is zero in $\mathbf{G}_2$. We have $\Pr[\mathbf{Succ}_2] = 1/2$.

Combining all the probabilities finishes the proof. □

As for the IND-CCA security, since the FO transformation is not changed, Theorem 4.4 is still applicable to the IND-CCA security of CNTR.KEM.

E.4 Concrete Hardness and Parameter Selection

The parameter sets of CNTR are given in Table 7, where those in red are the recommended parameters which are also given in Table 9. Though the parameters in red are marked as recommended, we believe the other parameter sets are still very useful in certain application scenarios. n and q are the ring parameters. q_2 is the compression modulus w.r.t. the RLWR problem. Recall that we fix the message space modulus $p = 2$ and the underlying cyclotomic polynomial $\Phi(x) = x^n - x^{n/2} + 1$, which are omitted in Table 7. Ψ and Ψ_r are the probability distribution which are set to be B_η or U_k, where B_η is the centered binomial distribution w.r.t. the integer η and U_k is the uniform distribution over $[-k, k] \cap \mathbb{Z}$. The public key sizes $|pk|$, ciphertext sizes $|ct|$ and B.W. (bandwidth, $|pk| + |ct|$) are measured in terms of bytes. "Sec.C" and "Sec.Q" mean the estimated core-SVP security level expressed in bits in the classical and quantum settings respectively, where both the types

Table 7: Parameter sets of CNTR

Schemes	n	q	q_2	(Ψ, Ψ_r)	$\lvert pk\rvert$	$\lvert ct\rvert$	B.W.	NTRU (Sec.C, Sec.Q)	RLWR (Sec.C, Sec.Q)	δ
CNTR-512	512	3457	2^9	(B_3, U_1)	768	576	1344	(118,107)	(118,107)	2^{-86}
	512	3457	2^{10}	(B_5, U_3)	768	640	1408	(127,115)	(123,111)	2^{-141}
	512	3457	2^{10}	(B_5, U_4)	768	640	1408	(127,115)	(128,116)	2^{-94}
CNTR-768	768	3457	2^{10}	(B_3, B_3)	1152	960	2112	(192,174)	(186,169)	2^{-269}
	768	3457	2^{10}	(B_3, U_2)	1152	960	2112	(192,174)	(190,172)	2^{-237}
	768	3457	2^{10}	(B_3, U_3)	1152	960	2112	(192,174)	(199,181)	2^{-158}
CNTR-1024	1024	3457	2^{10}	(B_3, B_3)	1536	1280	2816	(269,244)	(261,236)	2^{-195}
	1024	3457	2^{10}	(B_3, U_2)	1536	1280	2816	(269,244)	(265,241)	2^{-171}
	1024	3457	2^{10}	(B_3, U_3)	1536	1280	2816	(269,244)	(277,252)	2^{-113}

Table 8: Gate-count estimate of CNTR parameters. q_2 and (Ψ, Ψ_r) are referred to section E.4. d is the optimal lattice dimension for the attack. b is the BKZ blocksize. b' is the sieving dimension accounting for "dimensions for free". Gates and memory are expressed in bits. The last column means the required log(gates) values by NIST.

Schemes	q_2	(Ψ, Ψ_r)	d	b	b'	log(gates)	log(memory)	log(gates) by NIST
CNTR-512	2^9	(B_3, U_1)	981	385	349	143.7	88.2	143
	2^{10}	(B_5, U_3)	1025	461	420	164.8	103.4	
	2^{10}	(B_5, U_4)	1025	481	439	170.4	107.4	
CNTR-768	2^{10}	(B_3, B_3)	1498	671	618	224.6	145.3	207
	2^{10}	(B_3, U_2)	1496	684	630	228.1	147.8	
	2^{10}	(B_3, U_3)	1489	718	662	237.5	154.6	
CNTR-1024	2^{10}	(B_3, B_3)	1958	939	871	299.7	198.5	272
	2^{10}	(B_3, U_2)	1955	957	888	304.6	202.1	

Table 9: Comparison between CNTR and Saber. The column "Assumptions" refers to the underlying hardness assumptions. The column "Reduction" means that IND-CCA security is reduced to what kinds of CPA security, where "IND" ("OW") refers to indistinguishability (resp., one-wayness) and "RPKE" ("DPKE") refers to randomized (resp., deterministic) public-key encryptions. "Rings" refers to the underlying polynomial rings. The column "n" means the total dimension of algebraically structured lattices. "q" is the modulus. The public key sizes $|pk|$, ciphertext sizes $|ct|$, and B.W. (bandwidth, $|pk| + |ct|$) are measured in bytes. "Sec.C" and "Sec.Q" mean the estimated security expressed in bits in the classical and quantum setting respectively, which are gotten by the same methodology and scripts provided by NTRU KEM and Saber in NIST PQC Round 3, where we minimize the target values if the two hardness problems, say NTRU and RLWR, have different security values. The column "δ" indicates the error probabilities.

Schemes	Assumptions	Reduction	Rings	n	q	$\lvert pk\rvert$	$\lvert ct\rvert$	B.W.	(Sec.C, Sec.Q)	δ
CNTR (Ours)	NTRU, RLWR	IND-CPA RPKE	$\mathbb{Z}_q[x]/(x^n - x^{n/2} + 1)$	512	3457	768	640	1408	(123,111)	2^{-141}
				768	3457	1152	960	2112	(192,174)	2^{-158}
				1024	3457	1536	1280	2816	(265,241)	2^{-171}
Saber [14]	MLWR	IND-CPA RPKE	$\mathbb{Z}_q[x]/(x^{n/k} + 1)$ $k = 2, 3, 4$	512	8192	672	736	1408	(118,107)	2^{-120}
				768	8192	992	1088	2080	(189,172)	2^{-136}
				1024	8192	1312	1472	2784	(260,236)	2^{-165}

of NTRU attack and RLWR attack are considered. The last column "δ" indicates the error probabilities.

Remark that we estimate the classical and quantum core-SVP hardness security of the RLWR problem via the same scripts from Saber in NIST PQC Round 3 [14]. We also remark that CNTR enjoys a flexibility of parameter selections, but selecting these n's in Table 7 for CNTR is only for simplicity. One can also choose n from $\{576, 648, 864, 972, 1152, 1296\}$ which are integers of the form $3^l \cdot 2^e, l \geq 0, e \geq 1$. Note also that the plaintext message space of CNTR is $\{0, 1\}^{n/2}$, compared to the fixed message space $\{0, 1\}^{256}$ of Saber. The first parameter set of CNTR-512 w.r.t. $q_2 = 2^9$ dominates the smallest ciphertext size and bandwidth, whereas the third parameter set of CNTR-512 has the strongest hardness of lattice assumptions (say, NTRU and RLWR). They are practically applicable in certain application scenarios which are not much sensitive to error probability. In practice, each secret key will not be used for decryption more than 2^{80} times during its lifetime. In this case, the relatively higher error probabilities 2^{-86} and 2^{-94} do not undermine the actual security of these parameter sets in reality.

E.5 Refined Gate-Count Estimate

As for the quantum gates and space complexity, we use the same gate number estimation method as Saber in NIST PQC Round 3. The results of CNTR parameter sets are shown in Table 8. The data of CNTR-1024 w.r.t. $(q_2 = 2^{10}, \Psi = B_3, \Psi_r = U_3)$ is out of the range of the gate-count estimator, so we omit it.

E.6 Implementation and Benchmark

We provide the portable C implementation of our CNTR for the recommended parameter set of $(n = 768, q = 3457, q_2 = 2^{10}, \Psi = B_3, \Psi_r = U_3)$. Similar to CTRU, as for the prefix $ID(pk)$ of the public key h in CNTR, we use the first 33 bytes of the bit-packed NTT representation of h.

All the benchmark tests are run on an Intel(R) Core(TM) i7-10510U CPU at 2.3GHz (16 GB memory) with Turbo Boost and Hyperthreading disabled. The operating system is Ubuntu 20.04 LTS with Linux Kernel 4.4.0 and the gcc version is 9.4.0. The compiler flag is listed as follows: *-Wall -march=native -mtune=native -O3 -fomit-frame-pointer -Wno-unknown-pragmas*. We run the corresponding KEM algorithms for 10,000 times and calculate the average CPU cycles. The benchmark results of the schemes are shown in Table 10.

E.7 Comparisons with Saber

We mainly make a comparison between CNTR and Saber, that is based on the MLWR assumption. The source codes of Saber are taken from its Round 3 supporting materials. In addition, the FO transformation in Saber has been changed to $\mathrm{FO}^{\not\perp}_{ID(pk),m}$ as in [47]. The results are given in Table 9 and Table 10 for intuitive comparisons.

From Table 9, CNTR has smaller ciphertext sizes, stronger security and lower error probabilities for the three recommended parameter sets when compared to Saber. From Table 10, the Encaps process of CNTR-768 is still more efficient than that of Saber-768.

Table 10: CPU cycles of CNTR and Saber (in kilo cycles).

Schemes	KeyGen	Encaps	Decaps
CNTR-768	8.2×10^3	80.4	150.3
Saber-768 [14]	94.7	109.7	138.9