IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 13, 2020, accepted June 7, 2020, date of publication June 15, 2020, date of current version June 29, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3002223

Improved Ring LWR-Based Key Encapsulation Mechanism Using Cyclotomic Trinomials

SO HYUN PARK[1], SUHRI KIM[1], DONG HOON LEE[1], (Member, IEEE), AND JONG HWAN PARK[2]

[1]Graduate School of Information Security, Korea University, Seoul 02841, South Korea
[2]Department of Computer Science, Sangmyung University, Seoul 03016, South Korea

Corresponding author: Jong Hwan Park (jhpark@smu.ac.kr)

This work was supported in part by the Military Crypto Research Center funded by the Defense Acquisition Program Administration (DAPA) under Grant UD170109ED, and in part by the Agency for Defense Development (ADD).

ABSTRACT In the field of post-quantum cryptography, lattice-based cryptography has received the most noticeable attention. Most lattice-based cryptographic schemes are constructed based on the polynomial ring $R_q = \mathbb{Z}_q[x]/f(x)$, using a cyclotomic polynomial $f(x)$. Until now, the most preferred cyclotomic polynomials have been $x^n + 1$, where n is a power of two, and $x^n + \cdots + x + 1$, where $n + 1$ is a prime. The former results in the smallest decryption error size, but the choice of degree is limited. On the other hand, the latter gives rise to the largest decryption error size, but the choice of degree is very flexible. In this paper, we use a new polynomial ring $R_q = \mathbb{Z}_q/f(x)$ with a cyclotomic trinomial $f(x) = x^n - x^{n/2} + 1$ as an intermediate that combines the advantages of the other rings. Since the degree n is chosen freely as $n = 2^a 3^b$ for positive integers a and b, the choice of the degree n is moderate. Furthermore, since the error propagation is small in the middle of polynomial multiplication in the new ring, if the middle part is truncated and used, the decryption error size can be reduced. Based on these observations, we propose a new, practical key encapsulation mechanism (KEM) that is constructed over a ring with a cyclotomic trinomial. The security of our KEM is based on the hardness of ring learning-with-rounding (LWR) problems. With appropriate parameterization for the current 128-bit security model, we show that our KEM obtains shorter secret keys and ciphertexts, especially compared to the previous Ring-LWR-based KEM, Round5, with no error correction code. We then implement our KEM and compare its performance with that of several KEMs that were presented in the second round of the NIST PQC conference.

INDEX TERMS Cyclotomic trinomial, key encapsulation mechanism, lattice-based encryption, post-quantum cryptography, ring-LWR problem.

I. INTRODUCTION

As quantum computers advance, post-quantum cryptography has become one of the most demanding research topics. It has already been proven that, given idealized quantum computers, number-theoretic problems such as integer factorization and discrete logarithms can be solved in polynomial time, which means that currently employed cryptographic schemes based on these two hardness problems can also be broken in polynomial time. Accordingly, the National Institute of Standards and Technology (NIST) has launched the Post-Quantum Cryptography (PQC) Standardization to define new standards for cryptographic schemes including public key encryption (PKE) / key encapsulation mechanism (KEM), digital signature, and key-establishment protocols. In the second round of NIST PQC presentations in 2019, 12 out of 26 schemes were lattice-based, and 9 out of the 12 lattice-based schemes were PKE/KEM candidates. As this result indicates, lattices are the most prominent tool for designing cryptographic schemes at this point.

Such popularity can be traced back to Regev's work [1], which introduced the Learning With Errors (LWE) problem and proved its worst-case to average-case hardness reduction. In 2011, Lindner and Peikert [2] improved Regev's original PKE scheme and suggested a more efficient LWE-based PKE scheme in which a public key and ciphertexts are generated in the form of LWE instances. Since then, the improved construction by Lindner and Peikert has been adopted as a basic

The associate editor coordinating the review of this manuscript and approving it for publication was Cristina Rottondi.

design principle for most of the subsequent lattice-based PKE/KEM schemes. In 2012, Banerjee *et al.* [3] introduced a derandomized version of LWE, called the Learning With Rounding (LWR) problem. In contrast to the LWE problem, an LWR instance is generated by discarding some least significant bits through rounding operations without error sampling. Later, some LWR-based PKE/KEM schemes were suggested by [4], [5], with the advantage of shorter public keys and ciphertexts than in LWE-based constructions.

In 2010, Lyubashevsky *et al.* [6] introduced the ring variant of the LWE problem, called the Ring-LWE, and showed that it is hard to distinguish a Ring-LWE instance from a uniform one over cyclotomic polynomial rings. Despite the potential vulnerability inherent in an algebraic ring structure, PKE/KEM constructions based on the Ring-LWE problem can offer significant improvements in efficiency, such as much shorter public keys and ciphertexts. Similar to the LWR case, the Ring-LWR problem was defined in [3], and several PKE/KEM schemes [7], [8] have been proposed based on the Ring-LWR problem that have the advantages of shorter bandwidths and no error sampling. For this reason of efficiency improvements, most of the lattice-based PKE/KEM candidates presented in the second round of the NIST PQC conference were constructed based on the hardness of Ring-{LWE, LWR} problems or their variants.

A. MOTIVATION

Let $R_q = \mathbb{Z}_q[x]/f(x)$ be a polynomial ring for a cyclotomic polynomial $f(x)$ with a modulus q. Depending on the choice of polynomial $f(x)$, the computational speed of polynomial multiplication and the error propagation during decryption are determined. As a simpler explanation, consider an LWE-based PKE scheme in which the public key is given as $(\mathbf{a}, \mathbf{b} = \mathbf{a} \cdot \mathbf{s} + \mathbf{e_1}) \in R_q^2$, and $\mathbf{s} \in R_q$ is a secret key. Following [2], a ciphertext is generated as $(\mathbf{c_1} = \mathbf{a} \cdot \mathbf{r} + \mathbf{e_2}, \mathbf{c_2} = \mathbf{b} \cdot \mathbf{r} + \mathbf{e_3} + E(m)) \in R_q^2$ for some encoding method E of a message m. Now, the decryption algorithm performs $D(\mathbf{c_2} - \mathbf{c_1} \cdot \mathbf{s})$ for a decoding method D. Notice that the secret polynomials $\mathbf{s}, \mathbf{r}, \mathbf{e}_1, \mathbf{e}_2$, and $\mathbf{e}_3$ are sampled from distributions with small coefficients. The operation $\mathbf{c_2} - \mathbf{c_1} \cdot \mathbf{s}$ becomes $\mathbf{r} \cdot \mathbf{e_1} + \mathbf{e_3} - \mathbf{e_2} \cdot \mathbf{s} + E(m)$, where $\mathbf{r} \cdot \mathbf{e_1} + \mathbf{e_3} - \mathbf{e_2} \cdot \mathbf{s}$ is a decryption error. When E is the encoding method of Regev [1], the decryption succeeds only if $|\mathbf{r} \cdot \mathbf{e_1} + \mathbf{e_3} - \mathbf{e_2} \cdot \mathbf{s}| < q/4$ for each coefficient related to $E(m)$. Here, the point is that the size of the decryption error is dominated by the two polynomial multiplications $\mathbf{r} \cdot \mathbf{e_1}$ and $\mathbf{e_2} \cdot \mathbf{s}$ in R_q.

The most preferred polynomial ring for Ring-LWE or Ring-LWR is $R_q = \mathbb{Z}_q[x]/f(x)$, where $f(x) = x^n + 1$ and n is a power of two. This is because (1) the polynomial multiplication is simple and fast using the equation $x^n = -1$ in R_q, and (2) the decryption error is the smallest in that each coefficient of $\mathbf{r} \cdot \mathbf{e_1}$ and $\mathbf{e_2} \cdot \mathbf{s}$ is computed as the sum of n terms. The primary disadvantage is that the choice of n is critically restricted since the degree n must be a power of two, e.g., $n = 512, 1024, 2048$, etc. Moreover, from current lattice attacks based on sage module estimator[1] of [9], it has been shown that choosing $n = 512$ is not sufficient for providing the desired 128-bit security. To reach this security level, one might think to use $f(x) = x^{1024} + 1$ for the next cyclotomic polynomial, but the larger degree $n = 1024$ greatly increases the bandwidths of the resulting scheme. Another way is to increase the variance of the coefficients in the above secret polynomials, but in that case, the size of the decryption error also increases.

Another preferred polynomial ring is $R_q = \mathbb{Z}_q[x]/f(x)$, where $f(x) = x^n + \cdots + x + 1$ and $n + 1$ is a prime; this ring is used in Round5 [8]. In such a ring R_q, the polynomial multiplication is simple and fast using the equation $x^{n+1} = 1$ in R_q due to the fact that $x^{n+1} - 1 = (x-1)(x^n + \cdots + x + 1)$, and above all, there are many degrees to choose from, e.g., $n + 1 = 521, 523, 541, 547$, etc. To meet a specific security level, one can choose the degree n flexibly from among a dense set of n values. Despite these advantages, choosing the polynomial $x^n + \cdots + x + 1$ poses a serious problem of error propagation that significantly increases the size of the decryption error. Indeed, each coefficient of $\mathbf{r} \cdot \mathbf{e_1}$ and $\mathbf{e_2} \cdot \mathbf{s}$ in the decryption error becomes the sum of $2n - 2$ terms. Thus, for a similar polynomial degree, $f(x) = x^n + \cdots + x + 1$ almost doubles the error propagation compared to $x^n + 1$, resulting in a relatively higher decryption failure rate. Furthermore, in the case of Ring-LWR, all coefficients of $\mathbf{e_1}$ and $\mathbf{e_2}$ are sampled from a set with a uniform distribution. For instance, after a 3-bit rounding operation, the coefficients of $\mathbf{e_1}$ and $\mathbf{e_2}$ are *uniformly* distributed on $[-4, 3)$, which makes the error propagation larger than for Ring-LWE.

As mentioned above, the two preferred polynomial rings have pros and cons where the advantage of one is a disadvantage for the other. This motivates us to consider a new polynomial ring that properly combines their strengths. In this paper, we propose using a cyclotomic trinomial $f(x) = x^n - x^{n/2} + 1$, where $n = 2^a 3^b$ for positive integers a and b. For instance, possible choices of degree are $n = 486, 576, 648, 768$, etc, which shows that the set of possible degrees $\{n\}$ is sparser than for $x^n + \cdots + x + 1$ but denser than for $x^n + 1$. In terms of error propagation, we can truncate some of the coefficients of $\mathbf{r} \cdot \mathbf{e_1}$ and $\mathbf{e_2} \cdot \mathbf{s}$ in order to minimize decryption errors with the highest probability. This stems from the fact that, somewhat surprisingly, polynomial multiplication in a ring $R_q = \mathbb{Z}_q[x]/f(x)$ with a cyclotomic trinomial $f(x)$ affects the number of terms added to each coefficient of a multiplied polynomial. Figure 1 presents the number of terms added to a coefficient of a multiplied polynomial $c(x) = a(x) \cdot b(x)$ in each ring. Especially, in the case of the ring $R_q = \mathbb{Z}_q[x]/f(x)$ with $f(x) = x^n - x^{n/2} + 1$, we can see that the number of terms added increases linearly starting from the middle degree to the smallest degree. To minimize the size of the decryption error, we can therefore truncate the colored parts (in Figure 1) of $\mathbf{r} \cdot \mathbf{e_1}$ and $\mathbf{e_2} \cdot \mathbf{s}$, depending on the length of the encoded message $E(m)$, and use them to encrypt the message m. For

[1] https://bitbucket.org/malb/lwe-estimator/src/master/

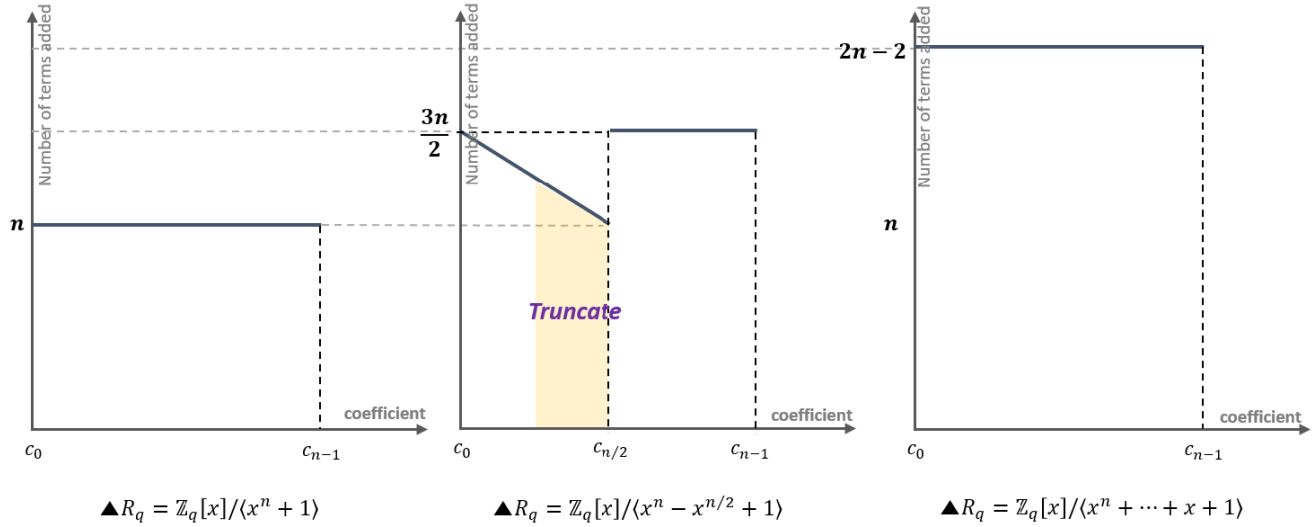


FIGURE 1. Number of terms $\{a_i b_j\}$ added into coefficients $\{c_k\}_{k=0}^{n-1}$, where $c_0 + c_1 x + \cdots + c_{n-1}x^{n-1} = (a_0 + a_1 x + \cdots + a_{n-1}x^{n-1}) \cdot (b_0 + b_1 x + \cdots + b_{n-1}x^{n-1})$ in each ring R_q.

instance, if $E(m)$ is 128 bits long, then each coefficient of the truncated $\mathbf{r} \cdot \mathbf{e_1}$ and $\mathbf{e_2} \cdot \mathbf{s}$ consists of the sum from terms n to $n + 128$. This is a slight increase over $x^n + 1$ but a very large decrease compared to $x^n + \cdots + x + 1$ in the number of terms added.

B. OUR CONTRIBUTION

Based on a new ring with the cyclotomic trinomials mentioned above, we propose an efficient KEM whose security relies on the hardness of the Ring-LWR problem. As always, we construct a PKE scheme that is secure against chosen-plaintext attacks (i.e., CPA-secure) and then apply the simpler version [10] of the Fujisaki-Okamoto transform to construct a KEM that is secure against chosen-ciphertext attacks (i.e., CCA-secure). To achieve the current 128-bit (classical) security level, we choose a set of parameters and compare the implementation result of our KEM with other comparable KEMs including NewHope [11], LAC [12], Round5 [8], Saber [5], and Kyber [13] in Section VI. The main features of our KEM are as follows:

- Cyclotomic trinomials on ring choice: our KEM is constructed based on a ring $R_q = \mathbb{Z}_q[x]/f(x)$ with $f(x) = x^{576} - 2^{288} + 1$ for $576 = 2^6 3^2$ for the 128-bit security level. Since the set of degrees is quite dense, we can also use cyclotomic trinomials with degrees $\{864 = 2^5 3^3, 1152 = 2^7 3^3\}$ to enhance security to the {192, 256}-bit security level, respectively.
- Negligible decryption failure rate: our KEM does not use an error correction code, but nevertheless the probability that a decryption fails (i.e., decryption failure rate) is negligible (i.e., 2^{-100}), using the truncation technique above. We provide a formula for the decryption failure rate that is calculated automatically, given our PKE/KEM parameters.
- Shorter size of ciphertexts: because of the new ring with cyclotomic trinomials and the truncation technique, we can set relatively small moduli for our KEM while preserving a negligible decryption failure rate. Indeed, our KEM uses moduli $(2^{11}, 2^8, 2^4)$, compared to $(2^{13}, 2^{10}, 2^2)$ in Saber and $(2^{13}, 2^9, 2^4)$ in Round5, without error correction code. The small moduli allow our KEM to obtain the shortest ciphertexts among the compared KEMs, as long as an error correction code is not used.
- Fixed-weight ternary distribution: following Round5 [8], two secret polynomials $\mathbf{r}$ and $\mathbf{s}$ in our KEM are sampled from a fixed-weight ternary distribution, where only non-zero coefficients are represented as an array called 'index.' We also adopt the index-based method to make polynomial multiplication in our new ring simple and fast.
- EMBLEM encoding method: we adopt the simple encoding method [14] to perform decryption with no rounding operation. Using the EMBLEM encoding, decoding (during decryption) is done by simply extracting the most significant bit from each coefficient, in contrast to the Regev encoding, which requires a rounding operation per each bit.

We show that polynomial multiplication in a ring $R_q = \mathbb{Z}_q[x]/f(x)$ with a cyclotomic trinomial $f(x) = x^n - x^{n/2} + 1$ can also be performed in a simple manner. Roughly speaking, let $\mathbf{a} = \Sigma_{i=0}^{n-1} a_i x^i$, $\mathbf{b} = \Sigma_{i=0}^{n-1} b_i x^i$, and $\mathbf{c} = \mathbf{a} \cdot \mathbf{b}$ in R_q. The first step is to compute $\mathbf{c}' = \Sigma_{i=0}^{2n-2} c_i x^i$ using the index-based multiplication method [], and the second step is to replace the terms from $c_n x^n$ to $c_{2n-2}x^{2n-2}$ by lower-degree terms. During the replacement process, we can use the relations $x^n = x^{n/2} - 1$ for the degrees $n, \ldots, \frac{3n}{2} - 1$ and $x^{3n/2} = -1$ for the degrees $\frac{3n}{2}, \ldots, 2n - 2$. Such polynomial multiplication

in R_q with cyclotomic trinomials is described as an algorithm in Section VI.

II. PRELIMINARIES

A. NOTATION

For $q \in \mathbb{N}$, $\mathbb{Z}_q$ denotes the set of all integers in $\{0, 1, \ldots, q-1\}$. For $r \in \mathbb{R}$, $\lfloor r \rceil$ is the nearest integer to r, rounded up in the case of a tie.

Suppose that $\mathbf{a}$ is a polynomial of degree $n-1$ whose coefficients are real-valued and $\mathbf{a}_i$ is the i-th coefficient of $\mathbf{a}$. Then, $\lfloor \mathbf{a} \rceil$ denotes a polynomial of degree n whose i-th coefficient is $\lfloor \mathbf{a}_i \rceil$. In addition, $\mathbf{a} \leftarrow \mathcal{P}$ denotes that $\mathbf{a}$ is an output of the algorithm $\mathcal{P}$, and $\mathbf{a} \xleftarrow{\$} U$ implies $\mathbf{a}$ is uniformly randomly chosen from a set U or sampled according to the distribution U. The hamming weight of $\mathbf{a}$ is the number of non-zero coefficients of $\mathbf{a}$. Define an algorithm $\mathcal{HWT}_n(\star, h)$ that has a seed $\star$ as an input and outputs a polynomial for which h out of n coefficients are -1 or 1 and the others are all zeros. For a bit string $b \in \{0,1\}^*$, $[b]_t$ is defined by the most significant t bits of b. Throughout this paper, we assume that q, p, and t are powers of 2 and that $t|p|q$. We represent polynomials as bold lowercase letters.

B. DEFINITIONS

1) PUBLIC KEY ENCRYPTION

A public key encryption (PKE) scheme consists of the following three algorithms: KeyGen, Encrypt, and Decrypt together with a message space $\mathcal{M}$.

- KeyGen(λ): The KeyGen algorithm takes as input a security parameter λ and outputs a public and secret key pair, (pk, sk).
- Encrypt(pk, m): The Encrypt algorithm takes as input the public key pk and a message $m \in \mathcal{M}$ and then outputs a ciphertext C.
- Decrypt(sk, C): The Decrypt algorithm takes as input the secret key sk and a ciphertext C and then outputs a message m or $\perp$ to indicate 'reject.'

Correctness. We say that a PKE is $(1-\epsilon)$-*correct* if the following condition holds: for all messages $m \in \mathcal{M}$,

$$\Pr\left[(pk, sk) \leftarrow \mathsf{KeyGen}(\lambda); C \leftarrow \mathsf{Encrypt}(pk, m) : \mathsf{Decrypt}(sk, C) = m\right] > 1 - \epsilon(\lambda),$$

where ϵ is a negligible function for the security parameter λ.

2) KEY ENCAPSULATION MECHANISM

A key encapsulation mechanism (KEM) consists of the following three algorithms: KeyGen, Encap, and Decap together with a key space $\mathcal{K}$.

- KeyGen(λ): The key generation algorithm takes as input a security parameter λ and outputs a public and secret key pair, (pk, sk).
- Encap(pk): The encapsulation algorithm takes as input the public key pk and then outputs a ciphertext C and a key $K \in \mathcal{K}$.
- Decap(sk, C): The decapsulation algorithm takes as input a secret key sk and a ciphertext C and then outputs a key K or $\perp$ to indicate 'reject.'

Correctness. We say that a KEM is $(1-\epsilon)$-*correct* if the following condition holds:

$$\Pr\left[(pk, sk) \leftarrow \mathsf{KeyGen}(\lambda); (C, K) \leftarrow \mathsf{Encap}(pk) : \mathsf{Decap}(sk, C) = K\right] > 1 - \epsilon(\lambda),$$

where ϵ is a negligible function for the security parameter λ.

3) IND-CPA SECURITY OF PKE

Let PKE=(KeyGen, Encrypt, Decrypt) be a public key encryption scheme. As the standard security notion for PKE, indistinguishability under chosen-plaintext attacks (IND-CPA) is defined via the following experiment between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$:

Experiment IND-CPA$^b_{\mathsf{PKE},\mathcal{A}}(\lambda)$:

1. *$\mathcal{C}$ runs $(pk, sk) \leftarrow$ KeyGen(λ) and gives pk to $\mathcal{A}$.*
2. *$\mathcal{A}$ outputs two messages (m_0, m_1) of the same length.*
3. *$\mathcal{C}$ computes $C \leftarrow$ Encrypt(pk, m_b) for a randomly chosen bit $b \in \{0,1\}$ and gives C to $\mathcal{A}$.*
4. *$\mathcal{A}$ outputs a bit b'. $\mathcal{C}$ returns 1 if $b = b'$ and otherwise returns 0 as the output of the game.*

The advantage of $\mathcal{A}$ for breaking the IND-CPA security of a PKE is defined as

$$\mathsf{Adv}^{\text{IND-CPA}}_{\mathsf{PKE},\mathcal{A}} = \left| \Pr\left[\mathbf{IND-CPA}^1_{\mathsf{PKE},\mathcal{A}}(\lambda) = 1\right] + \Pr\left[\mathbf{IND-CPA}^0_{\mathsf{PKE},\mathcal{A}}(\lambda) = 1\right] - 1 \right|.$$

We say that a PKE is IND-CPA secure if for any polynomial time adversary $\mathcal{A}$, we have $\mathsf{Adv}^{\text{IND-CPA}}_{\mathsf{PKE},\mathcal{A}} \leq \epsilon(\lambda)$, where ϵ is a negligible function for the security parameter λ.

4) IND-CCA SECURITY OF KEM

Let KEM = (KeyGen, Encap, Decap) be a key encapsulation mechanism. As the standard security notion for KEM, indistinguishability under chosen-ciphertext attacks (IND-CCA) is defined via the following experiment between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$:

Experiment IND-CCA$^b_{\mathsf{KEM},\mathcal{A}}(\lambda)$:

1. *$\mathcal{C}$ runs $(pk, sk) \leftarrow$ KeyGen(λ) and gives pk to $\mathcal{A}$.*
2. *$\mathcal{A}$ queries decapsulation oracle Decap$(sk, \cdot)$ with a ciphertext C.*
3. *$\mathcal{C}$ computes $(C^*, K_0^*) \leftarrow$ Encap(pk) and $K_1^* \xleftarrow{\$} \mathcal{K}$. Then $\mathcal{C}$ gives (C^*, K_b^*) to $\mathcal{A}$ for a randomly chosen bit $b \in \{0,1\}$.*
4. *$\mathcal{A}$ continues to query the decapsulation oracle but is not allowed to query the challenge ciphertext C^*.*
5. *Finally, $\mathcal{A}$ outputs a bit b'. $\mathcal{C}$ returns 1 if $b = b'$ and otherwise returns 0 as the output of the game.*

The advantage of $\mathcal{A}$ for breaking the IND-CCA security of KEM is defined as

$$\mathsf{Adv}^{\text{IND-CCA}}_{\mathsf{KEM},\mathcal{A}} = \left| \Pr\left[\mathbf{IND-CCA}^1_{\mathsf{KEM},\mathcal{A}}(\lambda) = 1\right] + \Pr\left[\mathbf{IND-CCA}^0_{\mathsf{KEM},\mathcal{A}}(\lambda) = 1\right] - 1 \right|.$$

We say that KEM is IND-CCA secure if for any polynomial time adversary $\mathcal{A}$, we have $\mathsf{Adv}^{\text{IND-CCA}}_{\mathsf{KEM},\mathcal{A}} \leq \epsilon(\lambda)$, where ϵ is a negligible function for the security parameter λ.

C. RING LEARNING WITH ROUNDING PROBLEMS

Let q, p, and t be positive integers that are represented as powers of 2 and that have the relation $t|p|q$. Let R_q, R_p, and R_t denote the rings $\mathbb{Z}_q[x]/\Phi_{3n}(x)$, $\mathbb{Z}_p[x]/\Phi_{3n}(x)$, and $\mathbb{Z}_t[x]/\Phi_{3n}(x)$, respectively, where $\Phi_{3n}(x)$ is the $3n$-th cyclotomic trinomial. More precisely, for any positive integers a and b, $\Phi_{3n}(x) = x^n - x^{n/2} + 1$ for $n = 2^a 3^b$.

With the above rings, we define two decisional Ring Learning With Rounding (RLWR) problems as a derandomized version of the Ring Learning With Errors (RLWE) problem [1], [6]. The first is the ordinary RLWR problem as proposed by [3], while the second is a slightly generalized version tailored to our security proofs.

Definition 1: Let n, q, p be positive integers such that $q > p$. Let R_q and R_p be polynomial rings constructed by $\Phi_{3n}(x)$, and let $\mathcal{D}_s$ be a distribution over R_q. A decisional RLWR problem $\mathsf{RLWR}_{n,1,q,p}(\mathcal{D}_s)$ is to distinguish uniformly random $(\mathbf{a}, \mathbf{u}) \in R_q \times R_p$ and $(\mathbf{a}, \mathbf{b} = \lfloor \frac{p}{q}\, \mathbf{a} \cdot \mathbf{s} \rceil) \in R_q \times R_p$, where $\mathbf{s}$ is sampled from $\mathcal{D}_s$. Then, the advantage of an adversary $\mathcal{A}$ in solving the decisional RLWR problem $\mathsf{RLWR}_{n,1,q,p}(\mathcal{D}_s)$ is defined as follows:

$$\mathsf{Adv}^{\mathsf{RLWR}}_{n,1,q,p}(\mathcal{A}) = \big|\Pr[\mathcal{A}(\mathbf{a}, \mathbf{b}) = 1] - \Pr[\mathcal{A}(\mathbf{a}, \mathbf{u}) = 1]\big|.$$

Definition 2: Let n, q, p, t be positive integers such that $q > p > t$. Let R_q, R_p, and R_t be polynomial rings constructed by $\Phi_{3n}(x)$, and let $\mathcal{D}_s$ be a distribution over R_q. A generalized version of the decisional RLWR problem $\mathsf{RLWR}_{n,2,q,p,t}(\mathcal{D}_s)$ is to distinguish two uniformly random pairs $\{(\mathbf{a}_1, \mathbf{u}_1) \in R_q \times R_p, (\mathbf{a}_2, \mathbf{u}_2) \in R_p \times R_t\}$ from $\{(\mathbf{a}_1, \mathbf{b}_1 = \lfloor \frac{p}{q}\, \mathbf{a}_1 \cdot \mathbf{s} \rceil) \in R_q \times R_p, (\mathbf{a}_2, \mathbf{b}_2 = \lfloor \frac{t}{p}\, \mathbf{a}_2 \cdot (\mathbf{s} \bmod p) \rceil) \in R_p \times R_t\}$, where $\mathbf{s}$ is sampled from $\mathcal{D}_s$. Then, the advantage of an adversary $\mathcal{A}$ in solving the generalized version of the decisional RLWR problem $\mathsf{RLWR}_{n,2,q,p,t}(\mathcal{D}_s)$ is defined as follows:

$$\mathsf{Adv}^{\mathsf{RLWR}}_{n,2,q,p,t}(\mathcal{A}) = \big|\Pr[\mathcal{A}(\{\mathbf{a}_i, \mathbf{b}_i\}_{i=1}^2) = 1] - \Pr[\mathcal{A}(\{\mathbf{a}_i, \mathbf{u}_i\}_{i=1}^2 = 1)]\big|.$$

The second RLWR problem can be viewed as a *module* version of two distinct RLWR problems, one from q to p and the other from p to t. Intuitively, the second RLWR problem fits more exactly into RLWR-based PKE/KEM constructions, and thus it is easy to understand our security analysis.

D. TRUNCATION FUNCTION

- **Truncation function.** Trunc is a function that takes as input a polynomial (in a ring) and outputs a truncated polynomial. Let $\mathbf{a} = \sum_{i=0}^{n-1} a_i x^i$ be a polynomial of degree $n-1$ and ℓ be a positive integer such that $\ell < n/2$. $\mathsf{Trunc}(\mathbf{a}, \ell)$ is defined as in Figure 2.
- **Fact.** For a constant $c \in \mathbb{Z}$, $c \times \mathsf{Trunc}(\mathbf{a}, \ell) = \mathsf{Trunc}(c\,\mathbf{a}, \ell)$.

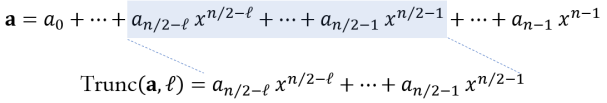


FIGURE 2. Truncation function.

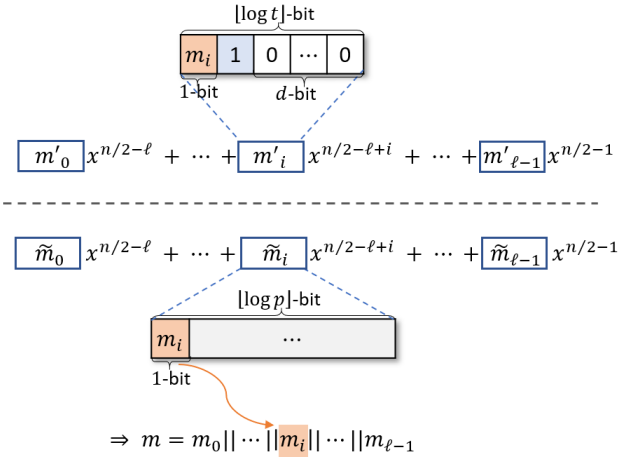


FIGURE 3. `R.Encode` (above) and `R.Decode` (below) functions.

E. EMBLEM ENCODING & DECODING METHOD

- **Encoding and decoding over rings.** Let $\mathcal{M} = \{0,1\}^\ell$ be a message space for a positive integer ℓ. Following [14], we define the encoding (decoding) function **R.Encode** (**R.decode**), which takes an ℓ-bit string (a truncated polynomial in a ring) as input and outputs a truncated polynomial (an ℓ-bit string). Let $\lfloor \log t \rfloor = d + 2$. For an ℓ-bit message m, a degree n, and a positive integer d, **R.encode** and **R.decode** work as follows:

▶ **R.encode**(m, n, d, ℓ)
1) Let m_i be the i-th bit of m for $i \in [0, \ell - 1]$.
2) Compute $m'_i = m_i || 1 || 0^d \in \mathbb{Z}_t$.
3) Output a polynomial $\mathbf{m} = \sum_{i=0}^{\ell-1} m'_i\, x^{n/2-\ell+i} \in R_t$.

▶ **R.decode**$(\mathbf{m}, n, d, \ell)$
1) $\mathbf{m} = \sum_{i=0}^{\ell-1} \tilde{m}_i\, x^{n/2-\ell+i} \in R_p$.
2) Let $m_i = [\tilde{m}_i]_1$ for $i \in [0, \ell - 1]$.
3) Output an ℓ-bit string $m = m_0 || \cdots || m_{\ell-1}$.

III. RLWR-BASED CPA-SECURE PKE

A. SCHEME

For the security parameter λ, the system parameter *params* is generated as follows: choose an integer n such that $n = 2^a 3^b$ for positive integers $a, b \in \mathbb{N}$ and modulus q, p, t as a power of 2 such that $t|p|q$. Also, choose positive integers d, ℓ, h, k such that $\log_2 t = d + 2$, ℓ is the bit-length of an encrypted message m, h is the number of non-zero elements among n coefficients, and k is the bit-length of a random $\star$ used in the $\mathcal{HWT}_n(\star, h)$ function. Then, *params* is given by $(n, d, \ell, h, k, q, p, t, R_q, R_p, R_t)$. Note that $R_q = \mathbb{Z}_q[x]/\Phi_{3n}(x)$, $R_p = \mathbb{Z}_p[x]/\Phi_{3n}(x)$, and $R_t = \mathbb{Z}_t[x]/\Phi_{3n}(x)$ for a $3n$-th cyclotomic trinomial $\Phi_{3n}(x) = x^n - x^{n/2} + 1$. It is

assumed that *params* is used for all algorithms in our PKE construction.

- *KeyGen*(λ).
  1) Choose a random polynomial $\mathbf{a} \in R_q$.
  2) Choose a random $\tilde{w} \in \{0,1\}^k$.
  3) Compute $\mathbf{x} \leftarrow \mathcal{HWT}_n(\tilde{w}, h)$.
  4) Compute $\mathbf{b} = \lfloor \frac{p}{q}\, \mathbf{a} \cdot \mathbf{x} \rceil \in R_p$.
  5) Output $pk = (\mathbf{a}, \mathbf{b})$ and $sk = (\mathbf{x})$.
- *Encrypt*($pk, m; w$). To encrypt a message $m \in \mathcal{M} = \{0,1\}^\ell$ under a random $w \in \{0,1\}^k$, the encryption algorithm proceeds as follows:
  1) $\mathbf{m} \leftarrow \mathtt{R.encode}(m, n, d, \ell)$.
  2) Compute $\mathbf{r} \leftarrow \mathcal{HWT}_n(w, h)$.
  3) Compute $\mathbf{u} = \lfloor \frac{p}{q}\, \mathbf{a} \cdot \mathbf{r} \rceil \in R_p$.
  4) Compute $\mathbf{v} = \mathsf{Trunc}(\lfloor \frac{t}{p}\, \mathbf{b} \cdot \mathbf{r} \rceil, \ell) + \mathbf{m}$.
  5) Output the ciphertext $C = (\mathbf{u}, \mathbf{v})$.
- *Decrypt*($sk, C = (\mathbf{u}, \mathbf{v})$).
  1) Compute $\tilde{\mathbf{m}} = \frac{p}{t}\, \mathbf{v} - \mathsf{Trunc}(\mathbf{u} \cdot \mathbf{x}, \ell)$.
  2) Output $m \leftarrow \mathtt{R.decode}(\tilde{\mathbf{m}}, n, d, \ell)$.

B. CORRECTNESS

From each of the rounding operations, we obtain the following equations:

$$\mathbf{e_b} = \mathbf{a} \cdot \mathbf{x} - \frac{q}{p} \lfloor \frac{p}{q}\, \mathbf{a} \cdot \mathbf{x} \rceil = \mathbf{a} \cdot \mathbf{x} - \frac{q}{p} \mathbf{b}, \tag{1}$$

$$\mathbf{e_u} = \mathbf{a} \cdot \mathbf{r} - \frac{q}{p} \lfloor \frac{p}{q}\, \mathbf{a} \cdot \mathbf{r} \rceil = \mathbf{a} \cdot \mathbf{r} - \frac{q}{p} \mathbf{u}, \tag{2}$$

$$\mathbf{e_v} = \mathbf{b} \cdot \mathbf{r} - \frac{p}{t} \lfloor \frac{t}{p}\, \mathbf{b} \cdot \mathbf{r} \rceil. \tag{3}$$

Notice that each coefficient of $\mathbf{e_b}$ and $\mathbf{e_u}$ is in $\left[-\frac{q}{2p}, \frac{q}{2p}\right)$ and that each coefficient of $\mathbf{e_v}$ is in $\left[-\frac{p}{2t}, \frac{p}{2t}\right)$.

Recall that q, p, and t are powers of 2 and that $t|p|q$, so that p/t can also be represented as a power of 2. During decryption, the polynomial $\tilde{\mathbf{m}}$ is computed as

$$\begin{aligned}\tilde{\mathbf{m}} &= \frac{p}{t}\, \mathbf{v} - \mathsf{Trunc}(\mathbf{u} \cdot \mathbf{x}, \ell) \\ &= \frac{p}{t}\, \mathsf{Trunc}(\lfloor \frac{t}{p}\, \mathbf{b} \cdot \mathbf{r} \rceil, \ell) + \frac{p}{t}\, \mathbf{m} - \mathsf{Trunc}(\mathbf{u} \cdot \mathbf{x}, \ell) \\ &= \mathsf{Trunc}(\frac{p}{t} \lfloor \frac{t}{p}\, \mathbf{b} \cdot \mathbf{r} \rceil - \mathbf{u} \cdot \mathbf{x}, \ell) + \frac{p}{t}\, \mathbf{m}.\end{aligned}$$

Since each coefficient of the polynomial $\mathbf{m}$ is of the form $m_i||1||0^d$, all coefficients of $\frac{p}{t}\, \mathbf{m}$ in the truncated polynomial are shifted left by $\log_2 \frac{p}{t}$ bits, i.e., $m_i||1||0^d 0^{\log_2 \frac{p}{t}}$ (which belongs to $\mathbb{Z}_p$). For a polynomial $\mathbf{e}$ in a ring R_q, we define $|\mathbf{e}|_i$ as the absolute value of the coefficient of the i-th term in $\mathbf{e}$. To correctly recover all message bits, the absolute value of each coefficient of the truncated polynomial $\mathsf{Trunc}(\frac{p}{t} \lfloor \frac{t}{p}\, \mathbf{b} \cdot \mathbf{r} \rceil - \mathbf{u} \cdot \mathbf{x}, \ell)$ must be smaller than $\frac{p}{t} \times 2^d$. This means that the following inequality should hold:

$$\left| \mathsf{Trunc}\left(\frac{p}{t} \lfloor \frac{t}{p}\, \mathbf{b} \cdot \mathbf{r} \rceil - \mathbf{u} \cdot \mathbf{x}, \ell\right) \right|_i < \frac{p}{t} \times 2^d,$$

for $i = \frac{n}{2} - 1, \ldots, \frac{n}{2} - \ell$.

Equivalently, in order to avoid the fractional notation of coefficients in $\mathsf{Trunc}(\frac{p}{t} \lfloor \frac{t}{p}\, \mathbf{b} \cdot \mathbf{r} \rceil - \mathbf{u} \cdot \mathbf{x}, \ell)$, the above inequality can be changed into the following:

$$\left| \mathsf{Trunc}\left(\frac{q}{p} \left(\frac{p}{t} \lfloor \frac{t}{p}\, \mathbf{b} \cdot \mathbf{r} \rceil - \mathbf{u} \cdot \mathbf{x}\right), \ell\right) \right|_i < \frac{q}{p} \times \frac{p}{t} \times 2^d \tag{4}$$

by shifting more $\lfloor \log_2 \frac{q}{p} \rfloor$ bits to the left. We also use the following fact:

$$\frac{q}{p}\, (\mathbf{b} \cdot \mathbf{r}) = (\frac{q}{p}\, \mathbf{b}) \cdot \mathbf{r}, \tag{5}$$

where the multiplication is carried out differently in R_p for the left-hand side and in R_q for the right-hand side. However, unless otherwise noted, parentheses will be omitted henceforth. Using equations (1)-(5), we now show the correctness of our PKE scheme.

Theorem 1: For $i = \frac{n}{2} - 1, \ldots, \frac{n}{2} - \ell$, let

$$\epsilon_i = \Pr\left[\left|\mathsf{Trunc}\left(-\mathbf{e_b} \cdot \mathbf{r} + \mathbf{e_u} \cdot \mathbf{x} - \frac{q}{p}\, \mathbf{e_v}, \ell\right)\right|_i < \frac{q}{t} \times 2^d\right],$$

where ℓ is the bit-length of an encrypted message and $d = \log_2 t - 2$. Let $\epsilon = 1 - \prod_i \epsilon_i$. Then, our (IND-CPA-secure) PKE scheme is $(1 - \epsilon)$ correct.

Proof: From the truncated polynomial in equation (4) above, we have the following equations:

$$\begin{aligned}&\mathsf{Trunc}\left(\frac{q}{p} \left(\frac{p}{t} \lfloor \frac{t}{p}\, \mathbf{b} \cdot \mathbf{r} \rceil - \mathbf{u} \cdot \mathbf{x}\right), \ell\right) \\ &= \mathsf{Trunc}\left(\frac{q}{t} \lfloor \frac{t}{p}\, \mathbf{b} \cdot \mathbf{r} \rceil - \frac{q}{p}\, \mathbf{u} \cdot \mathbf{x}, \ell\right) \\ &\overset{(3)}{=} \mathsf{Trunc}\left(\frac{q}{p}\, (\mathbf{b} \cdot \mathbf{r} - \mathbf{e_v}) - \frac{q}{p}\, \mathbf{u} \cdot \mathbf{x}, \ell\right) \\ &\overset{(5)(2)}{=} \mathsf{Trunc}\left(\frac{q}{p}\, (\mathbf{b} \cdot \mathbf{r} - \mathbf{e_v}) - (\mathbf{a} \cdot \mathbf{r} - \mathbf{e_u}) \cdot \mathbf{x}, \ell\right) \\ &\overset{(5)}{=} \mathsf{Trunc}\left(\left(\frac{q}{p}\, \mathbf{b} - \mathbf{a} \cdot \mathbf{x}\right) \cdot \mathbf{r} - \frac{q}{p}\, \mathbf{e_v} + \mathbf{e_u} \cdot \mathbf{x}, \ell\right) \\ &\overset{(1)}{=} \mathsf{Trunc}\left(-\mathbf{e_b} \cdot \mathbf{r} - \frac{q}{p}\, \mathbf{e_v} + \mathbf{e_u} \cdot \mathbf{x}, \ell\right).\end{aligned}$$

Thus, as long as $\left|\mathsf{Trunc}\left(-\mathbf{e_b} \cdot \mathbf{r} + \mathbf{e_u} \cdot \mathbf{x} - \frac{q}{p}\, \mathbf{e_v}, \ell\right)\right|_i < \frac{q}{t} \times 2^d$ simultaneously for $i = \frac{n}{2} - 1, \ldots, \frac{n}{2} - \ell$, the decryption succeeds. This means that the probability that the decryption succeeds becomes $\prod_i \epsilon_i$, which concludes the proof of Theorem 1. □

C. SECURITY

We now prove that the PKE scheme above is IND-CPA secure under the RLWR assumptions.

Theorem 2: For any adversary $\mathcal{A}$ against our PKE scheme, there exist distinguishers $\mathcal{B}$ and $\mathcal{C}$ such that

$$\mathsf{Adv}^{\text{IND-CPA}}_{\mathsf{PKE}}(\mathcal{A}) \le 2\, \mathsf{Adv}^{\mathsf{RLWR}}_{n,1,q,p}(\mathcal{B}) + 2\, \mathsf{Adv}^{\mathsf{RLWR}}_{n,2,q,p,t}(\mathcal{C}).$$

Proof: We prove the IND-CPA security of our PKE scheme through a sequence of hybrid games. For each $\mathbf{Game}_i$

for $i = 0, \ldots, 5$, S_i is defined as the event that an adversary $\mathcal{A}$ correctly guesses. **Game**$_0$ is the original game in which the ciphertext is an encryption of m_0, whereas in **Game**$_5$ the ciphertext is an encryption of m_1. Let $\mathcal{D}_i$ be the distribution of **Game**$_i$ for $i = 0, \ldots, 5$. It will be shown that $\mathcal{D}_0$ of **Game**$_0$ and $\mathcal{D}_5$ of **Game**$_5$ are computationally indistinguishable under two (decisional) RLWR assumptions from the view of $\mathcal{A}$.

▷ **Game 0.** Game 0 is the real game in which the public key is properly generated and the ciphertext is an encryption of m_0. Note that $\mathbf{m}_0 \leftarrow \mathtt{R.encode}(m_0, n, d, \ell)$. $\mathcal{D}_0$ is given as follows:

- $\mathcal{D}_0 = \{pk = (\mathbf{a}, \mathbf{b} = \lfloor \frac{p}{q}\mathbf{a} \cdot \mathbf{x} \rceil),$
  $C = (\lfloor \frac{p}{q}\, \mathbf{a} \cdot \mathbf{r} \rceil, \mathsf{Trunc}(\lfloor \frac{t}{p}\, \mathbf{b} \cdot \mathbf{r} \rceil, \ell) + \mathbf{m}_0)\}.$

▷ **Game 1.** In Game 1, b in the public key is replaced with a uniformly random polynomial in R_p. $\mathcal{D}_1$ is given as follows:

- $\mathcal{D}_1 = \{pk = (\mathbf{a}, \mathbf{b} \xleftarrow{\$} R_p),$
  $C = (\lfloor \frac{p}{q}\, \mathbf{a} \cdot \mathbf{r} \rceil, \mathsf{Trunc}(\lfloor \frac{t}{p}\, \mathbf{b} \cdot \mathbf{r} \rceil, \ell) + \mathbf{m}_0)\}.$

Therefore, $|\Pr[S_0] - \Pr[S_1]| \leq \mathsf{Adv}^{\mathsf{RLWR}}_{n,1,q,p}(\mathcal{B})$.

▷ **Game 2.** In Game 2, $\lfloor \frac{p}{q}\, \mathbf{a} \cdot \mathbf{r} \rceil$ and $\lfloor \frac{t}{p}\, \mathbf{b} \cdot \mathbf{r} \rceil$ are replaced with uniformly random polynomials in R_p and R_t, respectively. $\mathcal{D}_2$ is given as follows:

- Let $\mathbf{u} \xleftarrow{\$} R_p$ and $\mathbf{v} \xleftarrow{\$} R_t$.
- $\mathcal{D}_2 = \{pk = (\mathbf{a}, \mathbf{b} \xleftarrow{\$} R_p),$
  $C = (\mathbf{u}, \mathsf{Trunc}(\mathbf{v}, \ell) + \mathbf{m}_0)\}.$

Therefore, $|\Pr[S_1] - \Pr[S_2]| \leq \mathsf{Adv}^{\mathsf{RLWR}}_{n,2,q,p,t}(\mathcal{C})$.

▷ **Game 3.** In Game 3, the message encoding $\mathbf{m}_0$ is replaced with $\mathbf{m}_1 \leftarrow \mathtt{R.encode}(m_1, n, d, \ell)$. $\mathcal{D}_3$ is given as follows:

- Let $\mathbf{u} \xleftarrow{\$} R_p$ and $\mathbf{v} \xleftarrow{\$} R_t$.
- $\mathcal{D}_3 = \{pk = (\mathbf{a}, \mathbf{b} \xleftarrow{\$} R_p),$
  $C = (\mathbf{u}, \mathsf{Trunc}(\mathbf{v}, \ell) + \mathbf{m}_1)\}.$

Since $\mathbf{v}$ is uniformly random, $\mathcal{D}_2$ and $\mathcal{D}_3$ are identical. Therefore, $\Pr[S_2] = \Pr[S_3]$.

▷ **Game 4.** In Game 4, the ciphertext is restored to RLWR instances. $\mathcal{D}_4$ is given as follows:

- $\mathcal{D}_4 = \{pk = (\mathbf{a}, \mathbf{b} \xleftarrow{\$} R_p),$
  $C = (\lfloor \frac{p}{q}\, \mathbf{a} \cdot \mathbf{r} \rceil, \mathsf{Trunc}(\lfloor \frac{t}{p}\, \mathbf{b} \cdot \mathbf{r} \rceil, \ell) + \mathbf{m}_1)\}.$

Therefore, $|\Pr[S_3] - \Pr[S_4]| \leq \mathsf{Adv}^{\mathsf{RLWR}}_{n,2,q,p,t}(\mathcal{C})$.

▷ **Game 5.** In Game 5, b in the public key is restored to a RLWR instance. $\mathcal{D}_5$ is given as follows:

- $\mathcal{D}_5 = \{pk = (\mathbf{a}, \mathbf{b} = \lfloor \frac{p}{q}\mathbf{a} \cdot \mathbf{x} \rceil),$
  $C = (\lfloor \frac{p}{q}\, \mathbf{a} \cdot \mathbf{r} \rceil, \mathsf{Trunc}(\lfloor \frac{t}{p}\, \mathbf{b} \cdot \mathbf{r} \rceil, \ell) + \mathbf{m}_1)\}.$

Therefore, $|\Pr[S_4] - \Pr[S_5]| \leq \mathsf{Adv}^{\mathsf{RLWR}}_{n,1,q,p}(\mathcal{B})$.

It follows that

$$\begin{aligned}\mathsf{Adv}^{\text{IND-CPA}}_{\mathsf{PKE}}(\mathcal{A}) &= |\Pr[S_0] - \Pr[S_5]| \\ &\leq \sum_{i=0}^{4} \left|\Pr[S_i] - \Pr[S_{i+1}]\right| \\ &\leq 2\,\mathsf{Adv}^{\mathsf{RLWR}}_{n,1,q,p}(\mathcal{B}) + 2\,\mathsf{Adv}^{\mathsf{RLWR}}_{n,2,q,p,t}(\mathcal{C}),\end{aligned}$$

which concludes the proof of Theorem 2. □

IV. RLWR-BASED IND-CCA SECURE KEM

In this section, we describe an IND-CCA secure KEM in the (quantum) random oracle model over rings. We construct our KEM by applying a variant of the Fujisaki-Okamoto transformation [10], [15] to our IND-CPA secure PKE scheme.

A. SCHEME

Let $\mathsf{PKE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be an IND-CPA secure PKE scheme with a message space $\mathcal{M}$, a randomness space $\mathcal{R} = \{0,1\}^k$, where k is determined by the security parameter λ, a ciphertext space $\mathcal{C}$, and a key space $\mathcal{K}$. Let G and H be cryptographic hash functions such that $G : \mathcal{M} \to \mathcal{R}$ and $H : \mathcal{M} \times \mathcal{C} \to \mathcal{K}$. For the security parameter λ, our KEM then works as follows:

- *KeyGen*(λ).
  1) Generate $(pk', sk') \leftarrow \mathsf{PKE.KeyGen}(\lambda)$.
  2) Sample $s \xleftarrow{\$} \mathcal{M}$.
  3) $pk := pk'$ and $sk := (sk', s)$.
  4) Return (pk, sk).
- *Encap*(pk).
  1) Select a random $\delta \xleftarrow{\$} \mathcal{M}$.
  2) Compute $C = \mathsf{PKE.Enc}(pk, \delta; G(\delta))$.
  3) Compute $K = H(\delta, C)$.
  4) Output (K, C).
- *Decap*(sk, C).
  1) Parse $sk = (sk', s)$.
  2) Compute $\delta' = \mathsf{PKE.Dec}(sk', C)$.
  3) If $\mathsf{PKE.Enc}(pk, \delta'; G(\delta')) = C$,
  4) Return $K := H(\delta', C)$.
  5) Else, return $K := H(s, C)$.

Since the KEM above is constructed by our IND-CPA secure PKE scheme, the correctness of our KEM is straightforwardly obtained from that of the corresponding PKE scheme.

Theorem 3: If the underlying PKE scheme is $(1 - \epsilon)$-correct, then the above KEM is $(1 - \epsilon)$-correct.

B. SECURITY

Based on the results of [10], [15], we prove that our KEM is IND-CCA secure under two RLWR assumptions in the classical and quantum random oracle, respectively.

Theorem 4: (Theorems 3.2 and 3.4 in [15]). Assume the PKE is $(1-\epsilon)$ correct. For any IND-CCA adversary $\mathcal{B}$ against the KEM issuing at most q_G queries to random oracle G, q_H queries to random oracle H, and q_D queries to the decryption oracle, there exists an IND-CPA adversary $\mathcal{A}$ such that

$$\mathsf{Adv}^{\text{IND-CCA}}_{\mathsf{KEM}}(\mathcal{B}) \leq q_G \cdot \epsilon + \frac{2\,q_G + q_H + 1}{|\mathcal{M}|} + 3\mathsf{Adv}^{\text{IND-CPA}}_{\mathsf{PKE}}(\mathcal{A}),$$

and the running time of $\mathcal{A}$ is approximately that of $\mathcal{B}$.

Theorem 5: (Theorem 3 in [10]). Assume the PKE is $(1 - \epsilon)$ correct. For any IND-CCA adversary $\mathcal{B}$ against KEM issuing at most q_G queries to random oracle G, q_H queries

to random oracle H, and q_D queries to the decryption oracle, there exists an IND-CPA adversary $\mathcal{A}$ such that

$$\mathsf{Adv}^{\text{IND-CCA}}_{\text{KEM}}(\mathcal{B}) \leq 2q_H \frac{1}{\sqrt{|\mathcal{M}|}} + 4q_G\sqrt{\epsilon} + 2\sqrt{(q_G+q_H+1)\mathsf{Adv}^{\text{IND-CPA}}_{\text{PKE}}(\mathcal{A}) + 2\frac{(q_G+q_H+1)^2}{|\mathcal{M}|}},$$

and the running time of $\mathcal{A}$ is approximately that of $\mathcal{B}$.

V. PARAMETER SELECTION

A. ANALYSIS OF DECRYPTION FAILURE RATE

For an ℓ-bit encrypted message, Theorem 3 (and Theorem 1) shows that our KEM is $(1-\epsilon)$ correct, where $\epsilon = 1 - \prod_i \epsilon_i$ and

$$\epsilon_i = \Pr\left[\left|\mathsf{Trunc}\left(-\mathbf{e_b}\cdot\mathbf{r} + \mathbf{e_u}\cdot\mathbf{x} - \frac{q}{p}\mathbf{e_v}, \ell\right)\right|_i < \frac{q}{t}\times 2^d\right] \tag{6}$$

where q, p, and t are all powers of 2 such that $t|p|q$, $d = \log_2 t - 2$, and $i = \frac{n}{2}-1, \ldots, \frac{n}{2}-\ell$. We call the probability ϵ the decryption failure rate. In this section, we first show how the absolute size of each coefficient in the truncated polynomial $\mathsf{Trunc}\left(-\mathbf{e_b}\cdot\mathbf{r} + \mathbf{e_u}\cdot\mathbf{x} - \frac{q}{p}\mathbf{e_v}, \ell\right)$ can be calculated. To do this, with specific distributions of $\{\mathbf{e_b}, \mathbf{e_u}, \mathbf{e_v}\}$ and $\{\mathbf{r}, \mathbf{x}\}$, it is necessary to analyze how multiplication is performed in $R_q = \mathbb{Z}_q[x]/\Phi_{3n}(x)$, especially in the truncated part. We next present a formula that can automatically calculate the decryption failure rate, given the system parameters such as (n, d, ℓ, h, q, p, t).

1) DISTRIBUTIONS

- $\{\mathbf{e_b}, \mathbf{e_u}, \mathbf{e_v}\}$ distributions: from equations (1)-(3), we see that each coefficient of $\mathbf{e_b}$ and $\mathbf{e_u}$ is in $[-q/2p, q/2p)$ and each coefficient of $\mathbf{e_v}$ is in $[-p/2t, p/2t)$. We assume that coefficients of $\mathbf{e_b}$ and $\mathbf{e_u}$ are uniform in $[-q/2p, q/2p]$. The inclusion of the integer $q/2p$ can increase the size of coefficients, making the decryption failure rate larger than in the original (exclusion) case. Nevertheless, we assume the inclusion for simpler analysis when building a formula for the decryption failure rate.

- $\{\mathbf{r}, \mathbf{x}\}$ distributions: the two polynomials $\mathbf{x}$ and $\mathbf{r}$ are generated by the sampling function $\mathcal{HWT}_n(\star, h)$ for some random $\star$. This means that all of the n coefficients of $\mathbf{x}$ and $\mathbf{r}$ consist of $\{0, 1, -1\}$; among the n coefficients the number of 0 coefficients is $n-h$, and the number of $\{1, -1\}$ coefficients is h.

2) POLYNOMIAL MULTIPLICATION

To analyze the distributions of $\{\mathbf{e_b}\cdot\mathbf{r},\ \mathbf{e_u}\cdot\mathbf{x}\}$ in the above truncated polynomial, we need to show how multiplication is performed on the ring $R_q = \mathbb{Z}_q[x]/\Phi_{3n}(x)$, where $\Phi_{3n}(x) = x^n - x^{n/2} + 1$ is a $3n$-th cyclotomic trinomial. Generally, for two polynomials $\mathbf{a}, \mathbf{b} \in R_q$, such as $\mathbf{a} = \sum_{i=0}^{n-1} a_i x^i$ and $\mathbf{b} = \sum_{i=0}^{n-1} b_i x^i$, let $\mathbf{c} = \mathbf{a}\cdot\mathbf{b} = \sum_{i=0}^{n-1} c_i x^i \in R_q$. Conceptually,

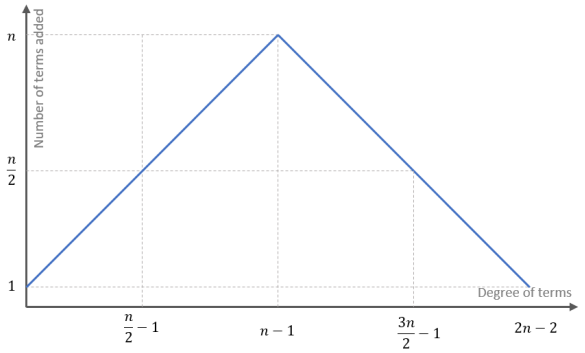


FIGURE 4. Relation between the number of $a_i b_j$ terms added (vertical) and the degree of the $c_k x^k$ term (horizontal).

before considering modular reductions by $\Phi_{3n}(x)$, we see that $\mathbf{c} = \sum_{i=0}^{2n-2} c_i x^i$, where each coefficient is represented as follows:

$$\begin{aligned}
c_0 &= a_0 b_0,\\
c_1 &= a_0 b_1 + a_1 b_0,\\
&\vdots\\
c_{n/2} &= a_0 b_{n/2} + \cdots a_{n/2} b_0,\\
&\vdots\\
c_{n-1} &= a_0 b_{n-1} + a_1 b_{n-2} + \cdots + a_{n-1} b_0,\\
&\vdots\\
c_{3n/2} &= a_{n/2+1} b_{n-1} + \cdots + a_{n-1} b_{n/2+1},\\
&\vdots\\
c_{2n-3} &= a_{n-2} b_{n-1} + a_{n-1} b_{n-2},\\
c_{2n-2} &= a_{n-1} b_{n-1}.
\end{aligned}$$

Figure 4 depicts the relation between the number of $a_i b_j$ terms added and the degree of $c_k x^k$ for $k = 0, \ldots, 2n-2$. We now perform modular reductions in R_q, using $\Phi_{3n}(x) = x^n - x^{n/2} + 1$, from the term $c_n x^n$ to $c_{2n-2}x^{2n-2}$. First, for $k = n, \ldots, \frac{3n}{2} - 1$, the coefficient c_k of the term $c_k x^k$ is added to the two terms $c_{k-n/2}x^{k-n/2}$ and $c_{k-n}x^{k-n}$, using the relation $x^n = x^{n/2} - 1$ in R_q. For instance, c_n (when $k = n$) is added to $c_{n/2}$ and c_0, and c_{n+1} (when $k = n+1$) is added to $c_{n/2+1}$ and c_1. These modular reductions are depicted in part (a) of Figure 5 in which the red dotted line is added into the blue solid line, leading to the purple line from degree 0 to $n-1$. Second, for $k = \frac{3n}{2}, \ldots, 2n-2$, the coefficient c_k of the term $c_k x^k$ is added to only one term $c_{k-3n/2}x^{k-3n/2}$, using the relation $x^{3n/2} = -1$ in R_q. For instance, $c_{3n/2}$ (when $k = 3n/2$) is added to c_0, and $c_{3n/2+1}$ (when $k = 3n/2+1$) is added to c_1. Similarly, these modular reductions are represented in part (b) of Figure 5, and the resultant number of terms added when computing $\mathbf{c} = \mathbf{a}\cdot\mathbf{b} = \sum_{i=0}^{n-1} c_i x^i \in R_q$ is shown in part (c) of Figure 5. Consequently, it can be seen that the coefficients $\{c_i\}$ from $i = \frac{n}{2} - 1$ to $\frac{n}{2} - \ell$ (for an ℓ-bit encrypted message) have the fewest addition

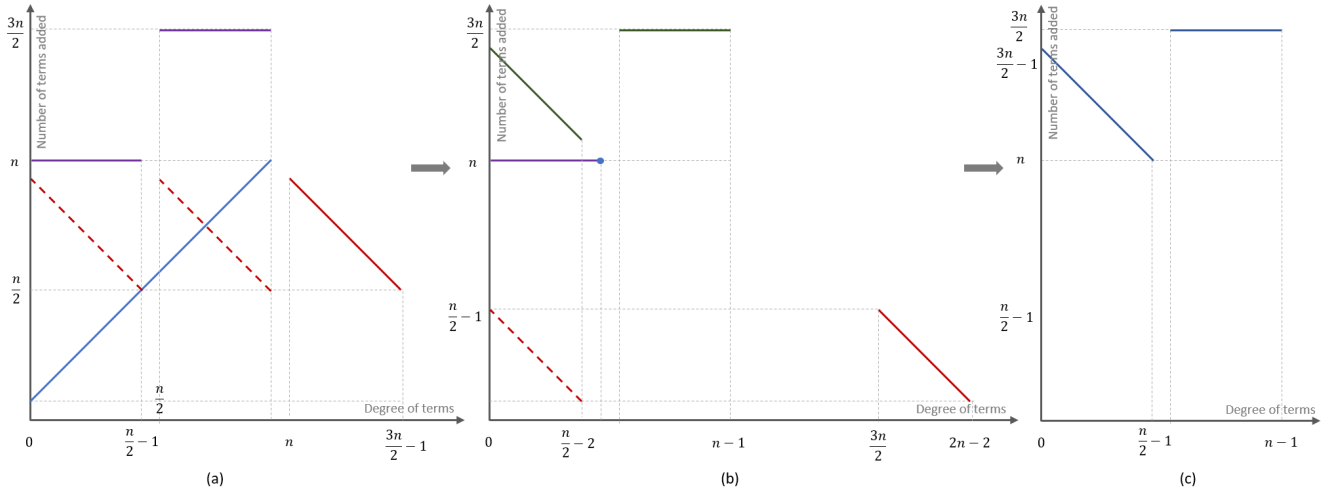


FIGURE 5. Modular reduction steps in ring $R_q = \mathbb{Z}_q[x]/\Phi_{3n}(x)$.

terms; this corresponds to the truncated part of our $\mathsf{Trunc}(\cdot)$ function.

We now can compute the distributions of $\{\mathbf{e_b} \cdot \mathbf{r},\ \mathbf{e_u} \cdot \mathbf{x}\}$ in the above truncated polynomial. Because the coefficient distributions of $\mathbf{e_b}$ and $\mathbf{r}$ are the same as those of $\mathbf{e_u}$ and $\mathbf{x}$, respectively, we focus on only one polynomial multiplication: $\mathbf{e_b} \cdot \mathbf{r}$. Let $\mathbf{c} = \mathbf{e_b} \cdot \mathbf{r} = \sum_{i=0}^{n-1} c_i x^i \in R_q$. Assuming the coefficients of $\mathbf{e_b}$ are uniform in $[-q/2p, q/2p]$ and $\mathbf{r}$ is sampled from $\mathcal{HWT}_n(w, h)$ for some random w, then c_i (in the truncated part) is a sum of $(\frac{3n}{2} - 1 - i)$ terms added for $i = \frac{n}{2} - 1$ to $\frac{n}{2} - \ell$. Among the summed terms in each c_i, only the h/n portion of the summed terms contributes to the sum of the inner products. As a result, c_i can be viewed as a sum of $(\frac{3n}{2} - 1 - i) \cdot \frac{h}{n}$ variables in $[-\frac{q}{2p}, \frac{q}{2p}]$, and by the Central Limit Theorem,[2] we see that each c_i (in the truncated part) converges to

$$N\Big(0, \Big(\frac{3n}{2} - 1 - i\Big) \cdot \frac{h}{n} \cdot \frac{q^2}{12p^2}\Big), \tag{7}$$

for $i = \frac{n}{2} - 1$ to $\frac{n}{2} - \ell$.

3) FORMULA FOR DECRYPTION FAILURE RATE

Recall that for a polynomial $\mathbf{a}$ in the ring R_q, we define $|\mathbf{a}|_i$ as the absolute value of the coefficient of the i-th term in $\mathbf{a}$. Theorem 3 (and Theorem 1) shows that the probability that the decryption succeeds is $\prod_i \epsilon_i$ for $i = \frac{n}{2} - 1, \ldots, \frac{n}{2} - \ell$, where

$$\epsilon_i = \Pr\Big[\big|\mathsf{Trunc}\big(-\mathbf{e_b} \cdot \mathbf{r} + \mathbf{e_u} \cdot \mathbf{x} - \frac{q}{p}\,\mathbf{e_v}, \ell\big)\big|_i < \frac{q}{t} \times 2^d\Big].$$

More precisely, ϵ_i is the probability that the i-th bit of an ℓ-bit message is recovered correctly. Now that we have the (distinct) distributions of the coefficients of $\{|\mathbf{e_b} \cdot \mathbf{r}|_i,\ |\mathbf{e_u} \cdot \mathbf{x}|_i,\ |q/p\ \mathbf{e_v}|_i\}$ for $i = \frac{n}{2} - 1, \ldots, \frac{n}{2} - \ell$, we can obtain a concrete bound for ϵ_i as follows:

$$\begin{aligned}
\epsilon_i &= \Pr\Big[\big|\mathsf{Trunc}\big(-\mathbf{e_b} \cdot \mathbf{r} + \mathbf{e_u} \cdot \mathbf{x} - \frac{q}{p}\,\mathbf{e_v}, \ell\big)\big|_i < \frac{q}{t} \times 2^d\Big] \\
&\geq \Pr\Big[\big|\mathsf{Trunc}\big(-\mathbf{e_b} \cdot \mathbf{r} + \mathbf{e_u} \cdot \mathbf{x}, \ell\big)\big|_i \\
&\quad + \big|\mathsf{Trunc}\big(\frac{q}{p}\,\mathbf{e_v}, \ell\big)\big|_i < \frac{q}{t} \times 2^d\Big] \\
&\geq \Pr\Big[\big|\mathsf{Trunc}\big(-\mathbf{e_b} \cdot \mathbf{r} + \mathbf{e_u} \cdot \mathbf{x}, \ell\big)\big|_i + \frac{q}{2t} < \frac{q}{t} \times 2^d\Big].
\end{aligned}$$

The last inequality holds because all coefficients of $\mathbf{e_v}$ are in $[-p/2t, p/2t)$, and thus $\big|\mathsf{Trunc}\big(\frac{q}{p}\,\mathbf{e_v}, \ell\big)\big|_i \leq \frac{q}{2t}$.

Assuming that all coefficients of $\{\mathbf{e_b}, \mathbf{e_u}\}$ are in $[-p/2t, p/2t]$ and $\{\mathbf{r}, \mathbf{x}\}$ are randomly chosen from $\mathcal{HWT}_n(\star, h)$ for some random $\star$, it is easy to see that from the equation (7), the distribution of $\big|\mathsf{Trunc}\big(-\mathbf{e_b} \cdot \mathbf{r} + \mathbf{e_u} \cdot \mathbf{x}, \ell\big)\big|_i$ converges to

$$N\Big(0,\ 2\Big(\frac{3n}{2} - 1 - i\Big)\frac{h}{n} \cdot \frac{q^2}{12p^2}\Big),$$

where $i = \frac{n}{2} - 1, \ldots, \frac{n}{2} - \ell$. Therefore, from the last inequality above (regarding ϵ_i), we continue to obtain

$$\begin{aligned}
\epsilon_i &\geq \Pr\Big[\big|\mathsf{Trunc}\big(-\mathbf{e_b} \cdot \mathbf{r} + \mathbf{e_u} \cdot \mathbf{x}, \ell\big)\big|_i < \frac{q}{t} \times 2^d - \frac{q}{2t}\Big] \\
&= \mathsf{erf}\Bigg(\frac{\frac{q}{t} \times 2^d - \frac{q}{2t}}{\sqrt{4\big(\frac{3n}{2} - 1 - i\big)\frac{h}{n} \cdot \frac{q^2}{12p^2}}}\Bigg).
\end{aligned}$$

In this case, the probability that the decryption succeeds is $\prod_i \epsilon_i$, and thus the decryption failure rate $\epsilon (= 1 - \prod_i \epsilon_i)$ is at most

$$\epsilon \leq 1 - \prod_i \mathsf{erf}\Bigg(\frac{\frac{q}{t} \times 2^d - \frac{q}{2t}}{\sqrt{4\big(\frac{3n}{2} - 1 - i\big)\frac{h}{n} \cdot \frac{q^2}{12p^2}}}\Bigg), \tag{8}$$

[2] We use the fact that the variance of the uniform distribution on $[-\frac{q}{2p}, \frac{q}{2p}]$ is $q^2/12p^2$.

TABLE 1. Comparison of CCA-secure KEMs over rings.

($\lambda = 128$)	NewHope	LAC$^\sharp$	Round5		Saber	Kyber	Ours*
			[I]	[II]†			
n	512	512	586	508	2×256	2×256	576
h	-	256	182	136	-	-	112
σ	2	-	-	-	2.24	1	-
q	12289	251	2^{13}	2^{10}	2^{13}	3329	2^{11}
p	-	-	2^9	2^7	2^{10}	2^{10}	2^8
t	-	-	2^4	2^4	2^2	2^3	2^4
ℓ	256	256	128	128	256	256	128
pk size (bytes)	928	544	676	461	672	800	608
sk size (bytes)§	1888(912)	1056(528)	276	201	1568(848)	1632(784)	156
C size (bytes)§	1120(1120)	712(648)	740	620	736(704)	736(688)	640
KeyGen (K cycles)	101	102	42	34	71	100	40
Encap (K cycles)	151	169	56	46	93	130	64
Decap (K cycles)	171	225	72	62	97	156	89
Decryption failure rate	2^{-213}	2^{-116}	2^{-157}	2^{-150}	2^{-120}	2^{-178}	2^{-100}
Security level (Quantum)	112 (101)	147(130)	131(119)	133(121)	126(115)	111(100)	128(116)
Hardness problem	RLWE	RLWE	RLWR	RLWR	MLWR	MLWE	RLWR

$\sharp$: In LAC, BCH code is used as an error correction code; $\dagger$: In Round5-**[II]**, XE5 is used as an error correction code; $\S$: The values in parentheses are the size of the secret keys and ciphertexts when each CCA-secure KEM is obtained via the simpler version [10] of the Fujisaki-Okamoto transform and 128-bit keys are encapsulated. $*$: The requirement of the decryption failure rate is to be negligible, and we considered about 2^{-100} as being a negligible probability. In addition, applying an error correction code to our KEM will significantly lower the bandwidth and decrease the decryption failure rate, which will be a subsequent work. In this paper, we show that just using a cyclotomic trinomial is able to improve the performance of our KEM.

where $i = \frac{n}{2} - 1, \ldots, \frac{n}{2} - \ell$. Notice that the upper bound in equation (8) can be calculated automatically, given the system parameters (n, d, ℓ, h, q, p, t).

B. PROPOSED PARAMETERS

In Table 1, we propose a set of parameters for our CCA-secure KEM described in Section IV, aiming at the 128-bit *classical* security level. We set $n = 576 = 2^6 3^2$ as the degree of a cyclotomic trinomial, $h = 112$ as the number of non-zero coefficients in the secret key $\mathbf{x}$ and a random polynomial $\mathbf{r}$, and $q = 2^{11}$, $p = 2^8$, and $t = 2^4$ as the modulo numbers. Under these parameters, we use the sage module estimator by Albrecht *et al.* [9] to estimate the security level for RLWR instances made from the public key and ciphertexts (without considering the security degradation in Theorem 4). According to this estimator, our proposed parameters give us 128-bit security.

Let $\ell = 128$, meaning that the underlying CPA-secure PKE scheme encrypts a 128-bit message. The public key of our KEM consists of two polynomials $pk = (\mathbf{a}, \mathbf{b}) \in R_q \times R_p$, where the random polynomial $\mathbf{a}$ can be generated from a 128-bit seed (using a pseudo-random function (PRF)) and $\mathbf{b}$ is a polynomial with $\log_2 p$-bit coefficients. The size of the public key is then $(256 + n \cdot \log_2 p)/8 = (256 + 576 \cdot 8) = 608$ bytes. The secret key of our KEM consists of $sk = (\mathbf{x}, s)$, where $\mathbf{x}$ is a polynomial generated from $\mathcal{HWT}_n(\star, h)$ for some random $\star \in \{0,1\}^k$ and s is an ℓ-bit random string. Using the index-based representation [16] of $\mathbf{x}$ in Round5, $\mathbf{x}$ is stored as an array of $(\mathsf{index}[0][0], \ldots, \mathsf{index}[\frac{h}{2} - 1][0], \mathsf{index}[0][1], \mathsf{index}[\frac{h}{2} - 1][1])$, where the array is filled with randomly generated values among $\{0, \ldots, n-1\}$ that indicate non-zero terms. (The algorithm for the $\mathcal{HWT}$ function will be described in Algorithm 1 in Section VI.) In that case, each entry is represented as a $\lceil \log_2 n \rceil$-bit string, and thus the size of the secret key is $(\ell + \lceil \log_2 n \rceil \cdot h)/8 = (128 + \lceil \log_2 576 \rceil \cdot 112)/8 = 156$ bytes. The ciphertext of our KEM consists of two polynomials $C = (\mathbf{u}, \mathbf{v})$, where $\mathbf{u} \in R_p$ and $\mathbf{v}$ is a truncated polynomial with $\log_2 t$-bit coefficients. Recall that the number of coefficients in $\mathbf{v}$ is ℓ. Thus, the size of a ciphertext is $(n \cdot \log_2 p + \ell \cdot \log_2 t)/8 = (576 \cdot 8 + 128 \cdot 4)/8 = 640$ bytes. Furthermore, by applying the above parameters to equation (8), we can see that the decryption failure rate of our KEM is up to 2^{-100}.

We notice that due to the density of polynomial degrees, we can provide additional parameter sets aiming for the 192-bit and 256-bit *classical* security levels. For instance, polynomial degrees $\{864 = 2^5 3^3, 1152 = 2^7 3^2\}$ can be chosen for the $\{192, 256\}$-bit classical security levels, respectively. These security levels are also measured based on sage module estimator by estimating the classical security of RLWR instances (without considering the security degradation in Theorem 5).

VI. IMPLEMENTATION

In this section, we describe subroutines characteristic of our implementation technique and compare the performance of our scheme with several CCA-secure KEMs over rings from among the NIST PQC second-round submissions.

A. TERNARY POLYNOMIAL GENERATION

In our KEM, the secret key $\mathbf{x}$ and a random polynomial $\mathbf{r}$ are generated from the function $\mathcal{HWT}_n(\star, h)$, given a k-bit random bit-string $\star$ as input. Algorithm 1

Algorithm 1: $\mathcal{HWT}_n(\star, h)$

Input : $\star \in \{0,1\}^k$
Output: index

1 int tmp$[n] = \{0\}$;
2 int index$[\frac{h}{2}][2]$;
3 int $i = 0, j = 0$;
4 $div \leftarrow$ the largest positive integer such that $div \cdot n < 2^{16}$;
5 **while** $i < h$ **do**
6 $z \leftarrow$ 16-bit random data;
7 //In the generation of 16-bit random data, we use
8 //SHAKE, a kind of hash function, with seed $\star$.
9 **if** $z < div \cdot n$ **then**
10 $j \leftarrow \lfloor \frac{z}{div} \rfloor$;
11 **if** tmp$[j]$ *is* 0 **then**
12 **if** i *is even* **then**
13 index$[\frac{i}{2}][0] = j$;
14 tmp$[i] = 1$;
15 **else**
16 index$[\frac{i}{2}][1] = j$;
17 tmp$[i] = -1$;
18 **end**
19 $i \leftarrow i + 1$;
20 **end**
21 **end**
22 **end**
23 free tmp;

Algorithm 2: Polynomial Multiplication

Input : index (instead of a ternary polynomial **a**),
$\mathbf{b} \in R_q$
Output: $\mathbf{c} \in R_q$

1 int $c_t[2n-1] = \{0,\}$;
2 int $c[n] = \{0,\}$;
3 // Compute $\mathbf{a} \cdot \mathbf{b} = \sum_{i=0}^{2n-2} c_t[i]x^i$
4 **for** $i = 0$ *to* $i = \frac{h}{2} - 1$ **do**
5 **for** $j = 0$ *to* $n - 1$ **do**
6 $c_t[j + \text{index}[i][0]] = c_t[j + \text{index}[i][0]] + b[j]$;
7 $c_t[j + \text{index}[i][1]] = c_t[j + \text{index}[i][1]] - b[j]$;
8 **end**
9 **end**
10 // Compute $\mathbf{a} \cdot \mathbf{b} = \sum_{i=0}^{n-1} c_t[i]x^i \pmod{\Phi_{3n}(x)}$
11 **for** $i = n$ *to* $\frac{3n}{2} - 1$ **do**
12 $c_t[i - \frac{n}{2}] = c_t[i - \frac{n}{2}] + c_t[i]$;
13 $c_t[i - n] = c_t[i - n] - c_t[i]$;
14 **end**
15 **for** $i = \frac{3n}{2}$ *to* $2n - 2$ **do**
16 $c_t[i - \frac{3n}{2}] = c_t[i - \frac{3n}{2}] - c_t[i]$;
17 **end**
18 **for** $i = 0$ *to* $n - 1$ **do**
19 $c[i] = c_t[i] \mod q$;
20 **end**
21 **return** $\mathbf{c} = \sum_{i=0}^{n-1} c[i]x^i$

describes the algorithm for $\mathcal{HWT}_n(\star, h)$, where the output is an array index $=$ (index[0][0], ..., index$[\frac{h}{2} - 1]$[0], index[0][1], ...,index$[\frac{h}{2} - 1]$[1]) representing the degrees of non-zero terms. The first half of the entries indicates the degrees of terms with coefficient 1, and the last half indicates the degrees of terms with coefficient -1. Let div be the largest positive integer such that $div \cdot n < 2^{16}$ with respect to the degree n of $\Phi_{3n}(x) = x^n - x^{n/2} + 1$. A procedure for generating z is initialized with a seed $\star$, and then 16-bit random strings are obtained by operating a kind of hash function, *SHAKE*. Note that two procedures initialized with the same seed generate two identical sequences of 16-bit strings. Every time z is newly generated, it is checked whether $z < div \cdot n$, and if so, a quotient through $\lfloor \frac{z}{div} \rfloor$ is assigned to an entry of index. Otherwise, z is re-generated. The reason for checking the inequality $z < div \cdot n$ is to find a random quotient (from 0 to $n - 1$) in an evenly distributed range $[0, div \cdot n)$. These processes continue until all h entries of index are filled.

B. POLYNOMIAL MULTIPLICATION

All polynomial multiplication in our KEM is performed in the form of $\mathbf{a} \cdot \mathbf{b}$, where $\mathbf{a}$ is a ternary polynomial (represented by an array index) and $\mathbf{b} \in R_q$ or R_p. Algorithm 2 presents the procedure of multiplying two polynomials $\mathbf{a} \cdot \mathbf{b}$ when taking index and $\mathbf{b} \in R_q$ as input. The whole procedure follows the same line of the polynomial multiplication explained in Section V. First, $\mathbf{a} \cdot \mathbf{b} = \sum_{k=0}^{2n-2} c_t[k]x^k$ is computed without $\Phi_{3n}(x)$ modular reduction, using the index. Since $c_t[k] = \sum_{i+j=k} a[i]b[j]$ conceptually, it is sufficient to consider only the h non-zero coefficients $\{a[i]\}$ that the index points to. This is equivalent to the computation in which $b[j]$ is added to or subtracted from $c_t[k]$ depending on whether $a[i] = 1$ or -1, respectively, under the relation $i + j = k$. Following this idea, lines 4-9 in Algorithm 2 show that each coefficient $b[j]$ of $\mathbf{b}$ is added to or subtracted from $c_t[k]$ for $k = 0, \ldots, 2n - 2$, depending on the h values in index. Second, the expanded $\mathbf{a} \cdot \mathbf{b} = \sum_{k=0}^{2n-2} c_t[k]x^k$ is reduced using the reduction polynomial $\Phi_{3n}(x) = x^n - x^{n/2} + 1$. As mentioned in Section V, this process can be done in two steps; for terms with degrees $k = n, \ldots, 3n/2 - 1$, we use the relation $x^n = x^{n/2} - 1$ in R_q, and for terms with degrees $k = 3n/2, \ldots, 2n - 2$, we use the relation $x^{3n/2} = -1$ in R_q. Lines 11-17 in Algorithm 2 show such a reduction process. Third, all coefficients of the resulting $\mathbf{a} \cdot \mathbf{b} = \sum_{k=0}^{n-1} c_t[k]x^k$ are reduced modulo q. We note that polynomial multiplication in R_p can be done in the same way, except that the modulus q in line 19 changes to modulus p.

C. COMPARISON TO PREVIOUS CCA-SECURE KEMs

Table 1 presents the performance result of our KEM and compares it to results from several CCA-secure KEMs [5], [8], [11]–[13] over rings. All implementations are performed

in C on an Intel Core i7-6700k running Ubuntu 18.04 LTS, and GNU GCC version 7.5.0 is used for the compilation. For comparison with the approximately 128-bit security level (NIST security category 1), we select the following parameter sets for each scheme; `NewHope-512`,[3] `LAC128`,[4] `R5ND_1KEM_0d`,[5] `LightSaber-KEM`,[6] and `KYBER512`.[7] Our implementation codes are available to https://github.com/RLWR-KEM/RLWR_KEM.

Regarding a cyclotomic reduction polynomial of the ring $R_q = \mathbb{Z}_q/f(x)$, NewHope, LAC, Saber, and Kyber use the polynomial $f(x) = x^n + 1$, where n is a power of 2, while Round5 uses the polynomial $f(x) = x^n + \cdots + x + 1$, where $n + 1$ is a prime. In contrast, our KEM uses the new trinomial $f(x) = x^n - x^{n/2} + 1$ with $n = 2^a 3^b$ for positive integers a and b. For choosing a reduction polynomial, $x^n + 1$ has a limited number of degrees, whereas $x^n + \cdots + x + 1$ provides a wider range of degrees. In comparison, with a moderate number of degrees $x^n - x^{n/2} + 1$ is an intermediate between the two reduction polynomials. Especially in the case of $f(x) = x^n + 1$, sage module estimator of [9] shows that the degree (or dimension) $n = 512$ is not sufficient for achieving 128-bit security, and this poses a problem in that choosing the next degree $n = 1024$ can cause the resulting KEMs to be inefficient.

In the case of LAC and NewHope using the reduction polynomial $x^{512} + 1$, LAC enhances the security by reducing the modulus to the 8-bit $q = 251$, which inevitably increases the decryption failure rate. Thus, LAC is required to additionally use an error correction code such as the BCH code [12]. In contrast, NewHope chooses to increase the modulus to the larger 14-bit $q = 12289$ and set the variance of the coefficients in the secret polynomials to be $\sigma^2 = 4$, but the obtained security is far below the 128-bit level even with a better decryption failure rate. On the other hand, Saber and Kyber are constructed based on the Module-{LWR, LWE} problems, respectively, where $x^{256} + 1$ is used as a basic module. The module technique has the same effect that multiples of 256 (such as 512, 768, 1024) can be selected as the degree of a reduction polynomial. However, as shown in Table 1, Saber and Kyber are slightly lacking in fully providing 128-bit security when choosing the polynomial degree $n = 2 \times 256$. In order to compensate for the security loss, a larger $n = 3 \times 256 = 768$ can be chosen, but the sizes of the public key, secret key, and ciphertext then also increase. Otherwise, when using $n = 2 \times 256$ as it is, one can increase the variance σ^2 of the coefficients in the secret polynomials. In that case, however, there is a disadvantage in that the decryption failure rate increases.

With the polynomial $x^n + \cdots + x + 1$, Round5 offers a wider range of parameter selection than the other compared KEMs. However, as mentioned before, this polynomial gives the same error propagation effect as using a polynomial of a degree of approximately $2n$ when compared to the polynomial $x^n + 1$. To alleviate this error propagation problem, Round5-[I] has to set larger moduli $q = 2^{13}$ and $q = 2^9$, although a larger number of a polynomial degree $n = 586$ is also chosen. Table 1 shows that the larger moduli of Round5-[I] yield a larger public key, secret key, and ciphertext than those of our KEM. To reduce the larger bandwidth, Round5-[II] uses an error correction code called XEf, an f-bit error correcting code originally introduced in the HILA5 [17]. In comparison, our KEM provides the shortest bandwidth among the comparable KEMs, unless an error correction code is also used. This is even true when NewHope, Saber, and Kyber are modified to encapsulate 128-bit keys and the simpler version [10] of the Fujisaki-Okamoto transform is applied to them.

The secret polynomials in NewHope, Saber, and Kyber are sampled from centered binomial distributions, whereas the secret polynomials in LAC, Round5, and our KEM are sampled from fixed-weight ternary distributions. As shown in [18], sampling secret polynomials from a ternary distribution makes the ciphertext and public key smaller than when sampling from a centered binomial distribution. In addition, it also enables a secret polynomial to be represented with an array of length h. As shown in Table 1, this allows our KEM to achieve shorter keys than the comparable KEMs. Moreover, NewHope and Kyber use the NTT (number theoretic transform) [19] to speed up polynomial multiplication in a ring R_q. To do this, they use the modulus $q = 12289$ in NewHope and $q = 3329$ in Kyber. LAC uses the modulus $q = 251$ due to its security analysis. However, from our implementation results in Table 1, it can be seen that KEMs with a power-of-two modulus (including Round5, Saber, and ours) perform much faster than those KEMs that do not have a power-of-two modulus.

VII. CONCLUSION

In this paper, we proposed a new KEM whose security is based on Ring-LWR problems. Unlike previous Ring-LWR-based KEMs, our scheme has a clear distinction in that it is constructed using a ring with a cyclotomic trinomial $x^n - x^{n/2} + 1$ where $n = 2^a 3^b$ for positive integers a and b. While two polynomial rings $\mathbb{Z}_q[x]/x^n + 1$ and $\mathbb{Z}_q[x]/x^n + \cdots + x + 1$ have been used to construct several KEMs that are presented in the second round NIST PQC conference, the two rings have distinct advantages and disadvantages with respect to the density of parameter space and decryption failure rate. The new ring with cyclotomic trinomial, however, has advantages in both and can be viewed as an intermediate that provides a moderate set of parameter choices and also quite low decryption failure rates. Moreover, our experiment showed that using cyclotomic trinomial rings results in not only smaller-sized ciphertexts and secret keys, but also comparable cpu cycles, compared to previous Ring-LWE/LWR-based KEMs with no error correction codes. In addition,

[3] https://newhopecrypto.org/data/NewHope_2019_07_10.pdf
[4] https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions
[5] https://round5.org/
[6] https://www.esat.kuleuven.be/cosic/pqcrypto/saber/
[7] https://pq-crystals.org/kyber/index.shtml

we presented a formula for computing decryption failure rates in cyclotomic trinomial rings for given security parameters. We expect that our KEM over cyclotomic trinomial rings is used as a reference data for future works.

REFERENCES

[1] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," in *Proc. 37th Annu. ACM Symp. Theory Comput. (STOC)*, 2005, pp. 84–93.

[2] R. Lindner and C. Peikert, "Better key sizes (and attacks) for LWE-based encryption," in *Topics in Cryptology* (Lecture Notes in Computer Science), vol. 6558. Berlin, Germany: Springer, 2011, pp. 319–339.

[3] A. Banerjee, C. Peikert, and A. Rosen, "Pseudorandom functions and lattices," in *Advances in Cryptology* (Lecture Notes in Computer Science), vol. 7237. Berlin, Germany: Springer, 2012, pp. 719–737.

[4] J. H. Cheon, D. Kim, J. Lee, and Y. Song, "Lizard: Cut off the tail! A practical post-quantum public-key encryption from LWE and LWR," in *Security and Cryptography for Networks* (Lecture Notes in Computer Science), vol. 11035. Cham, Switzerland: Springer, 2018, pp. 160–177.

[5] J. D'Anvers, A. Karmakar, S. S. Roy, and F. Vercauteren, "Saber: Module-LWR based key exchange, CPA-secure encryption and CCA-secure KEM," in *Progress in Cryptology* (Lecture Notes in Computer Science), vol. 10831. Cham, Switzerland: Springer, 2018, pp. 282–305.

[6] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," in *Advances in Cryptology* (Lecture Notes in Computer Science), vol. 6110. Berlin, Germany: Springer, 2010, pp. 1–23.

[7] J. Lee, D. Kim, H. Lee, Y. Lee, and J. H. Cheon, "RLizard: Post-quantum key encapsulation mechanism for IoT devices," *IEEE Access*, vol. 7, pp. 2080–2091, 2019.

[8] H. Baan, S. Bhattacharya, S. R. Fluhrer, O. García-Morchón, T. Laarhoven, R. Rietman, M. O. Saarinen, L. Tolhuizen, and Z. Zhang, "Round5: Compact and fast post-quantum public-key encryption," in *Advances in Cryptology* (Lecture Notes in Computer Science), vol. 11505. Cham, Switzerland: Springer, 2019, pp. 83–102.

[9] M. R. Albrecht, B. R. Curtis, A. Deo, A. Davidson, R. Player, E. W. Postlethwaite, F. Virdia, and T. Wunderer, "Estimate all the LWE, NTRU chemes!," in *Security and Cryptography for Networks* (Lecture Notes in Computer Science), vol. 11035. Cham, Switzerland: Springer, 2018, pp. 351–367.

[10] H. Jiang, Z. Zhang, and Z. Ma, "Tighter security proofs for generic key encapsulation mechanism in the quantum random oracle model," in *Post-Quantum Cryptography* (Lecture Notes in Computer Science), vol. 11505. Cham, Switzerland: Springer, 2019, pp. 227–248.

[11] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe, "Post-quantum key exchange—A new hope," in *Proc. USENIX Secur. Symp.*, 2016, pp. 327–343.

[12] X. Lu, Y. Liu, Z. Zhang, D. Jia, H. Xue, J. He, and B. Li, "LAC: Practical ring-LWE based public-key encryption with byte-level modulus," in *Proc. IACR*, 2018, p. 1009.

[13] J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehle, "CRYSTALS–kyber: A CCA-secure Module-Lattice-Based KEM," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, Apr. 2018, pp. 353–367.

[14] M. Seo, S. Kim, D. H. Lee, and J. H. Park, "EMBLEM: (R)LWE-based key encapsulation with a new multi-bit encoding method," *Int. J. Inf. Secur.*, vol. 2019, Jul. 2019, doi: 10.1007/s10207-019-00456-9.

[15] D. Hofheinz, K. Hövelmanns, and E. Kiltz, "A modular analysis of the Fujisaki-Okamoto transformation," in *Theory of Cryptography* (Lecture Notes in Computer Science), vol. 10677. Cham, Switzerland: Springer, 2017, pp. 341–371.

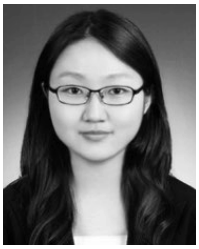
[16] H. Baan, S. Bhattacharya, J. H. Cheon, S. Fluhrer, O. Garcia-Morchon, P. Gorissen, T. Laarhoven, R. Player, R. Rietman, M.-J. O. Saarinen, L. Tolhuizen, J. L. Torre-Arce, and Z. Zhang. (2019). *Round5: KEW and PKE Based on (Ring) Learning With Rounding*. [Online]. Available: https://round5.org/doc/Round5_Submission022020.pdf

[17] M. O. Saarinen, "HILA5: On reliability, reconciliation, and error correction for ring-LWE encryption," in *Selected Areas in Cryptography* (Lecture Notes in Computer Science), vol. 10719. Cham, Switzerland: Springer, 2017, pp. 192–212.

[18] S. Bhattacharya, O. García-Morchón, R. Player, and L. Tolhuizen, "Achieving secure and efficient lattice-based public-key encryption: The impact of the secret-key distribution," in *Proc. IACR*, 2019, p. 389.

[19] T. Güneysu, T. Oder, T. Pöppelmann, and P. Schwabe, "Software speed records for lattice-based signatures," in *Post-Quantum Cryptography* (Lecture Notes in Computer Science), vol. 7932. Berlin, Germany: Springer, 2013, pp. 67–82.

SO HYUN PARK received the B.S. degree from the Department of Mathematics, Korea University, Seoul, South Korea, in 2019. She is currently pursuing the M.S. degree with the Graduate School of Information Security, Korea University. Her research interests include post-quantum cryptography and cryptographic protocols.



SUHRI KIM received the B.S. degree from the Department of Mathematics, Korea University, Seoul, South Korea, in 2014, and the M.S. and Ph.D. degrees with the Graduate School of Information Security, Korea University, in 2016 and 2020, respectively. Her research interests include post-quantum cryptography and efficient computations for isogeny-based cryptosystems.

DONG HOON LEE (Member, IEEE) received the B.S. degree from the Department of Economics, Korea University, Seoul, South Korea, in 1985, and the M.S. and Ph.D. degrees in computer science from The University of Oklahoma, USA, in 1988 and 1992, respectively. Since 1993, he has been with the Faculty of Computer Science and Information Security, Korea University. He is currently a Professor and the Director of the Graduate School of Information Security, Korea University. His research interests include cryptographic protocol, applied cryptography, functional encryption, software protection, mobile security, vehicle security, and ubiquitous sensor network (USN) security.

JONG HWAN PARK received the B.S. degree from the Department of Mathematics, Korea University, Seoul, South Korea, in 1999, and the M.S. and Ph.D. degrees with the Graduate School of Information Security, Korea University, in 2004 and 2008, respectively. From 2009 to 2011, he was a Research Professor with Kyung Hee University. From 2011 to 2013, he was a Research Professor with Korea University. Since 2013, he has been an Assistant Professor with the Department of Computer Science, Sangmyung University, Seoul. His research interests include functional encryption, broadcast encryption, authenticated encryption, and various cryptographic protocols.