Improved Meet-LWE Attack via Ternary Trees

Eunmin Lee[1], Joohee Lee[1⋆], Yuntao Wang[2]

[1] Sungshin Women's University, Seoul, Republic of Korea
{20211089, jooheelee}@sungshin.ac.kr
[2] The University of Electro-Communications, Japan
y-wang@uec.ac.jp

Abstract. The Learning with Errors (LWE) problem with its variants over structured lattices has been widely exploited in efficient post-quantum cryptosystems. Recently, May [59] suggests the Meet-LWE attack, which poses a significant advancement in the line of work on the Meet-in-the-Middle approach to analyze LWE with ternary secrets.
In this work, we generalize and extend the idea of Meet-LWE by introducing ternary trees, which result in diverse representations of the secrets. More precisely, we split the secrets into three pieces with the same dimension and expand them into a ternary tree to leverage the increased representations to improve the overall attack complexity. We carefully analyze and optimize the time and memory costs of our attack algorithm exploiting ternary trees, and compare them to those of the Meet-LWE attack. With asymptotic and non-asymptotic comparisons, we observe that our attack provides improved estimations for all parameter settings, including those of the practical post-quantum schemes, compared to the Meet-LWE attack. We also evaluate the security of the Round 2 candidates of the KpqC competition which aims to standardize post-quantum public key cryptosystems in the Republic of Korea, and report that the estimated complexities for our attack applied to SMAUG-T are lower than the claimed for some of the recommended parameters.

Keywords: Learning with Errors, Meet-LWE, Meet-in-the-Middle, KpqC Competition

1 Introduction

There have been rapid advances in Post-Quantum Cryptography (PQC) since the National Institute of Standards and Technology (NIST) launched a standardization project for post-quantum Key Encapsulation Mechanisms (KEM) and digital signatures [60]. Remarkably, three lattice-based schemes Kyber [19], Dilithium [35], and Falcon [39] out of 4 in total are selected as standards at the end of the third round. Likewise, standardization efforts for PQC have been made in South Korea, conducting a KpqC competition since 2022 [24].

⋆ corresponding author

In these circumstances, the Learning with Errors (LWE) problem with its variants over structured lattices [62,56,51,21] is currently, out of question, one of the richest sources of efficient post-quantum cryptosystems including Kyber [19] and Dilithium [35]. Moreover, various cryptographic primitives have been proposed based on LWE due to its versatility and fast operations [22,32,55]. An LWE instance with m samples is given as $(A, \vec{b}) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$ where $A \in \mathbb{Z}_q^{m \times n}$ is uniformly sampled, and a small $\vec{s} \in \mathbb{Z}_q^n$ and a small error $\vec{e} \in \mathbb{Z}_q$ satisfying $A \cdot \vec{s} = \vec{b} + \vec{e} \mod q$ exist. The (search) LWE problem aims to find the secret vector $\vec{s} \in \mathbb{Z}_q^n$ for given an LWE instance $(A, \vec{b})$, where $\vec{b} - A \cdot \vec{s}$ is sufficiently small.

Though the theoretical hardness of LWE for large n is grounded by the reduction to worst-case lattice hard problems, the parameterization for n, q, and the error distributions remains complex to achieve practical LWE-based cryptographic schemes while guaranteeing concrete security against all existing attacks. Many recent practical constructions have opted for extremely short secrets and errors with bounded max-norms to enhance efficiency. For instance, some popular signature schemes such as BLISS [34], GLP [40] and NTRU-type encryption schemes such as NTRU [44], NTRU Prime [16] and NTRU+ [1] utilize binary or ternary secrets and errors. Additionally, the state-of-the-art fully homomorphic encryption (FHE) schemes like BGV [22] and CKKS [32] employ ternary secrets. Moreover, several schemes [44] make use of sparse ternary or sparse binary secrets with fixed Hamming weights for efficiency and specific functionalities, such as handling the decryption failure rates in NTRU-type schemes where the decryption failure implies fatal attacks [47,33] and enabling the bootstrapping in FHEs [27,30].

However, while fruitful ideas and results for tackling the LWE problem have been suggested so far [9], the cryptanalytic hardness of LWE with sparse or ternary secrets is less understood. Exploiting the features of small secrets, the combinatorial attack is considered as one of the most natural strategies, and it yields a better attack complexity when combined with the lattice reduction techniques. Nevertheless, the Meet-in-the-Middle (MitM) approach proposed by Odlyzko [44] has long been the best combinatorial attack, resulting in an attack complexity of $S^{0.5}$, where S is the size of the search space for the secret key.

Recently, Alexandar May introduced a pioneering MitM combinatorial attack named Meet-LWE with an improved asymptotic complexity of $S^{0.25}$ [59]. Meet-LWE combines the Howgrave-Graham's representation technique [46] with a tree-based list construction for the secrets and their locality sensitive hash values. In a high-level overview, May represents the LWE secret $\vec{s} \in \{0, \pm 1\}^n := \vec{s}_1 + \vec{s}_2$ where $\vec{s}_1, \vec{s}_2 \in \{0, \pm 1\}^n$ have the same dimension n, and leverage the plural representations of $\vec{s}$ by guessing some of the coordinates of $\vec{e}$ to reduce the list construction complexity for $\vec{s}$. Refer to Section 2.4 for more details.

Table 1: Comparison on Time Complexity Estimations of May's Meet-LWE Attack (REP-2) and Our Attack for Various Schemes

Schemes	(n, q, w)	May [bit]	Ours [bit]
NTRU-Encrypt [44]	$(509, 2048, 254)$	$227 = 189+38$	$\mathbf{192} = 173+19$
	$(677, 2048, 254)$	$273 = 231+42$	$\mathbf{214} = 190+24$
	$(821, 4096, 510)$	$378 = 318+60$	$\mathbf{346} = 318+28$
NTRU Prime [16]	$(653, 4621, 288)$	$272 = 229+42$	$\mathbf{232} = 213+19$
	$(761, 4591, 286)$	$301 = 258+43$	$\mathbf{242} = 218+24$
	$(857, 5167, 322)$	$338 = 291+47$	$\mathbf{273} = 247+26$
BLISS I+II [34]	$(512, 12289, 154)$	$187 = 163+24$	$\mathbf{151} = 136+15$
GLP I [40]	$(512, 8383489, 342)$	$225 = 206+20$	$\mathbf{217} = 194+23$
NTRU+ [1]	$(576, 3457, 288)$	$263 = 228+36$	$\mathbf{221} = 200+21$
	$(768, 3457, 384)$	$349 = 302+47$	$\mathbf{287} = 261+26$
	$(864, 3457, 432)$	$392 = 339+53$	$\mathbf{319} = 288+31$
	$(1152, 3457, 576)$	$519 = 448+71$	$\mathbf{433} = 397+36$
SMAUG-T [2]	$(512, 1024, 100)$	$144 = 124+21$	$\mathbf{122} = 98+24$
	$(512, 1024, 132)$	$167 = 147+20$	$\mathbf{147} = 132+16$
	$(768, 2048, 151)$	$214 = 192+21$	$\mathbf{182} = 161+21$
	$(1280, 2048, 160)$	$283 = 255+29$	$\mathbf{231} = 210+21$

1.1 Our Contribution

In this paper, we concentrate on the setting of LWE with ternary secrets and errors of which each component lies in $\{0, \pm 1\}$, where the limited number of samples are given ($m = n$). We generalize the Meet-LWE attack and suggest an improved combinatorial attack by changing the way of constructing lists for candidates of solutions and their locality sensitive hash values on top of the Meet-LWE strategy.

Roughly speaking, the core idea underlying Meet-LWE is to reduce the list sizes for secrets by introducing the representation technique, with a factor of $\lfloor \log_q R \rfloor$ on each level of the tree where R is the number of representations on that level. On the other side, it costs a guessing complexity on each level multiplied by the list construction complexity, so the attack complexity, when increasing tree levels or switching the representation strategies for larger R's, converges quickly in their approach.

Our attack generalizes the state-of-the-art Meet-LWE attack in a new dimension by extending it to operate over a ternary tree three-armed on each level except for the tree's top level. Looking more closely, we split the secret $\vec{s}$ on each level of the tree into a sum of three vectors $\vec{s} := \vec{s}_1 + \vec{s}_2 + \vec{s}_3$ of the same dimension n instead of sum of two vectors, which brings us not only a drastically increased diversity of representations but also a decreased guessing complexity in our analysis. This gives us an avalanche effect spreading from the bottom level to the top level of the tree, greatly reducing the overall time complexity. Our algorithm achieves the asymptotic complexity ranging from $S^{0.21}$ to $S^{0.22}$ de-

pending on the ratio between the Hamming weights and dimension of the LWE secrets, which defeats that of the Meet-LWE attack. Also, our attack and cost estimation do not rely on the conservative assumptions such as Core-SVP in the lattice reductions, or the heuristics such as Geometric Series Assumption.

We have analyzed and optimized the attack complexities and show that our approach gives better time complexity in the regime of practical parameters compared to the May's attack. Our attack provides the reduced complexity estimation compared to Meet-LWE's best results with an extent from 8 to 65 bits for the parameters of schemes in [43,16,34,40] as shown in Table 1.

We also evaluate the security for the two Round 2 candidates using the ternary LWE problem, NTRU+ [1,49] and SMAUG-T [2,29], in the on-going KpqC competition which aims to standardize post-quantum public-key cryptosystems in the Republic of Korea [24]. Both NTRU+ and SMAUG-T have four parameter sets according to the security levels I, I, III, and V presented in order respectively in 10-th to 17-th rows of Table 1. For SMAUG-T, by exploiting the sparse ternary secrets in their scheme, we achieve the reduced complexity estimations even lower than the claimed security (estimated without the conservative core-SVP model) for {TiMER, SMAUGT192, SMAUGT256} parameter sets corresponding to the first, third, fourth rows of SMAUG-T parameters in Table 1, which introduces a necessity to revise the security claims. We remark that it does not imply these parameters need to be replaced, but our complexity results can serve as another criteria to estimate the security and set the parameters for achieving the claimed security accordingly. For NTRU+, our attack yields better complexities than Meet-LWE for all parameters in a large extent. We believe our results would be useful for the standardization process for PQC and for future work to evaluate the security of ternary secret LWE in more depths.

1.2 Related Work

Lattice Reductions. Lattice reduction algorithms assess the concrete difficulty of lattice problems such as NTRU or LWE by converting them to (approximate) Shortest Vector Problem (SVP) or Closest Vector Problem (CVP) and then resolving them by finding the secret short vectors from the transformed bases. The BKZ reduction algorithm and its variants [28,11] iteratively apply the LLL reduction [52] and an enumeration algorithm with fixed or flexible block sizes.

Alternatively, sieve algorithms [5] can replace the enumeration SVP oracle. Notably, Becker et al. [14] proposed a LD sieve algorithm with locality-sensitive filter (LSF), which is capable to solve SVP in $2^{0.292n+o(n)}$ time for an n-dimensional lattice. The LD Sieve algorithm shows the best asymptotic time complexity for SVP and is popular for evaluating the security parameters of certain lattice-based cryptosystems.

Lattice Attacks. Several lattice-based attacks on the LWE problem have been studied by framing LWE as a particular instance of the bounded distance decoding (BDD) problem on a q-ary lattice using Babai's nearest plane (NP)

algorithm [12] or Kannan's embedding technique [48] in the attack strategies so-called "decoding attack" and "primal attack". BDD involves finding the closest lattice vector $A\vec{s}$ to a given target vector $\vec{b}$ within a reasonable bound of $\|\vec{e}\|$ in a q-ary lattice. "Dual attack" is proposed to solve the decision-LWE problem where one is asked to distinguish whether a given sample comes from an LWE instance of $(A, \vec{b}) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$ or a uniform distribution of $(U, \vec{u}) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$ [4]. The basic idea of dual attacks is to use short vectors in the dual (or say, the orthogonal complement) of the lattice to detect the statistical distance between a new generated sample and the uniform one.

To estimate the concrete hardness of LWE, Albrecht et al. use several time models from fplll, enumeration, and sieve to evaluate the concrete hardness of LWE by invoking several algorithms such as BKW, SIS, decoding attack, etc [9]. In Asiacrypt 2017, Albrecht et al. revisited the hardness of LWE using the so-called "2016 estimate" and a BKZ time model adopting sieving as SVP oracle [8]. Some other theoretical analyses for the hardness of LWE are given as lattice-based attack [53,54,64], and BKW type combinatorial attacks [50,41]. We remark that the BKW type combinatorial attacks [50,41] require specific setup with large number of samples and superpolynomial modulus, so it is not suitable for our setting with $m = n$.

Recently, Ducas and Pulles have identified significant contradictions between the underlying heuristics assumed in these dual attacks and both formal theorems and well-established heuristics, as demonstrated through thorough theoretical analysis and extensive experiments in [36].

Conversely, Pouly and Shen have addressed these gaps by presenting a provable dual attack on the LWE problem that does not rely on statistical assumptions [61]. Their approach utilizes a simplified yet rigorous method rooted in geometric analysis rather than heuristic statistical models. The techniques include Monte Carlo Markov Chain discrete Gaussian sampling to estimate the complexity of the attack on specific parameter sets, with a particular focus on the Kyber encryption scheme. However, the cost is not yet competitive compared to the state-of-the-art dual attack with heuristic assumptions, since they do not cover the modulus switching in their formal analysis.

Meet-in-the-Middle Attacks. The idea of Odlyzko's MitM attack on ternary NTRU was initially introduced in [44], which is also a classical combinatorial approach used to address the LWE problem, particularly with binary or ternary LWE keys. The fundamental idea of Odlyzko's MitM algorithm involves rewriting the given LWE instance $(A, \vec{b}) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$ as $A_1\vec{s}_1 = \vec{b} - A_2\vec{s}_2 + \vec{e}$. Here, $A = (A_1|A_2) \in \mathbb{Z}_q^{m\times n/2} \times \mathbb{Z}_q^{m\times n/2}$, and $\vec{s} = (\vec{s}_1|\vec{s}_2) \in \{0, \pm 1\}^{n/2} \times \{0, \pm 1\}^{n/2}$ with the same Hamming weight for $\vec{s}_1$ and $\vec{s}_2$, respectively. Then, we randomly sample $\vec{s}_1$ and $\vec{s}_2$, and store them into two lists labeled by a locality sensitive hash function that inputs $A_1\vec{s}_1$ and $\vec{b} - A_2\vec{s}_2$, respectively. As there is only a small difference $\vec{e}$ between $A_1\vec{s}_1$ and $\vec{b} - A_2\vec{s}_2$, the goal is to recover the secret $\vec{s}$ by finding a collision between the two lists. Odlyzko's MitM algorithm runs in time $S^{0.5}$, where S represents the size of the exponential search space. This approach has been pivotal for combinatorial attacks on LWE and has inspired

various improvements and adaptations, including quantum variants [18,45] and hybrid attacks that combine with lattice reduction algorithms (see Section 7).

It is notable that, in the CRYPTO 2007 paper by Howgrave and Graham [46], they utilized a full length of $\vec{s}_1, \vec{s}_2 \in \{0, \pm 1\}^n$. This resulted in a higher computational cost compared to Odlyzko's MitM algorithm, as indicated by May's analysis [59]. Nonetheless, Howgrave-Graham's algorithm inspired a new MitM attack known as the Meet-LWE algorithm [59]. The Meet-LWE algorithm shows a significant reduction in complexity, with asymptotic runtime of $S^{0.25}$ and achieving non-asymptotic complexities of around $S^{0.3}$ for certain parameter sets.

1.3 Paper Organization.

In Section 2, we introduce lattice-based hard problems, LWE, with bounded secrets and errors, and then recall May's Meet-LWE attack along with its useful definition and Lemma. In Section 3, we introduce our algorithm extended on top of the Meet-LWE attack, operating on ternary trees in a high-level idea. Then, we instantiate our attack with various strategies, REP-0 in Section 4, REP-1-0 and REP-1-1 in Section 5, and calculate the time complexities for each of the representation strategies. In Section 6, we present the optimized complexities of our attack and compare them with May's. We also estimate the security of the Round 2 candidates of the KpqC competition using our attack in the same section. Finally, in Section 7, we discuss the applicability of our attack with hybrid attacks.

2 Preliminaries

2.1 Notations

For a positive integer q, $\mathbb{Z}_q$ denotes the ring of integers mod q. We use $\mathbb{Z}_q^n$ for $\mathbb{Z}_q$ in n-dimension. We denote by $\mathcal{T}^n = \mathbb{Z}_q^n \cap \{-1, 0, 1\}^n$ the set of n-dimensional ternary vectors. We use $x \leftarrow \mathcal{S}$ to denote the sampling x from the distribution $\mathcal{S}$. For a real number r, $\lfloor r \rfloor$ is the floor function that outputs the greatest integer less than or equal to r. We denote by $\ell(\cdot) : \mathbb{Z}_q^n \to \{0,1\}^n$ the locality sensitive hash function used in the Odlyzko's attack where an i-th component of $\ell(\vec{x})$ is 0 if an i-th component of $\vec{x}$ is in a range $[0, \lfloor q/2 \rfloor - 1)$, and 1 otherwise. We denote by $\pi_r : \mathbb{Z}_q^n \to \mathbb{Z}_q^r$ the projection map onto the first r coordinates.

2.2 Useful Definition and Lemma

For computing the attack complexity, we use some useful definition and lemma from [59].

Definition 1 ([59]). *The Hamming weight $w := \Sigma_{s_i \neq 0} 1$ is the number of non-zero components in $\vec{s} = (s_1, ..., s_n) \in \mathbb{F}_q^n$. The set of ternary weight-w vectors in n dimension is denoted by*

$$\mathcal{T}^n(w/2) = \{\vec{s} \in \mathcal{T}^n \mid \vec{s} \text{ has } w/2 \ (\pm 1)\text{-entries each}\}.$$

Any rounding is omitted for simplicity. We define a relative weight $0 \leq \omega \leq 1$ satisfying $w = \omega \cdot n$. As noted in [59], $\omega \in [1/3, 2/3]$ yields optimal parameter choices in NTRU-type cryptosystems, e.g., $\omega = \frac{3}{8}$ for $n = 677$ in NTRU.

Lemma 1 ([59]). *Let $\mathcal{C} = \{c_1, ..., c_h\}$ be a set of numbers with cardinality h and let $\Sigma_{i=1}^{h} k_i = 1$. The number of vectors $\vec{s} \in \mathcal{C}^n \cap \mathbb{Z}_q^n$ with $k_i n$ many c_i-entries such that $\Sigma_{i=1}^{h} k_i = 1$ can be computed as follows.*

$$\binom{n}{k_1 n, ..., k_h n} \approx 2^{H(k_1,...,k_h)n},$$

where $H(k_1, ..., k_h) := \Sigma_{i=1}^{h} k_i \log_2 \left(\frac{1}{k_i}\right)$.

To prove the lemma, we use the Stirling approximation. We note that $k_h = 1 - \sum_{i=1}^{h-1} k_i$ is automatically determined by k_i's for $1 \leq i \leq h-1$. Hence, we define the following formula for notational convenience.

$$\binom{n}{k_1 n, ..., k_{(h-1)} n, \cdot} := \binom{n}{k_1 n, ..., k_h n},$$

$$H(k_1, ..., k_{(h-1)}, \cdot) := H(k_1, ..., k_{(h-1)}, k_h).$$

2.3 LWE with Bounded Secrets and Errors

Let m, n and q be positive integers and $s \in \mathbb{Z}_q^n$ be a secret vector sampled from the secret distribution $\mathcal{S}$. Let χ be an error distribution over $\mathbb{Z}$. The Learning with Errors (LWE) distribution $A_{m,n,q,\chi}^{LWE}(\vec{s})$ is obtained by first sampling $\vec{a} \leftarrow \mathbb{Z}_q^n$ uniformly and $e \in \mathbb{Z} \leftarrow \chi$, computing $b = \langle \vec{a}, \vec{s} \rangle - e \mod q$, and then outputting $(\vec{a}, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ as a result. Given m samples $\{(\vec{a}_i, b_i = \langle \vec{a}_i, \vec{s} \rangle - e_i)\}_{i=1}^{m}$ from $A_{m,n,q,\chi}^{LWE}(\vec{s})$, the (search) LWE problem asks to recover the secret vector $\vec{s}$. We denote the LWE instance of m samples in a form $(A, \vec{b}) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$, where $A \leftarrow \mathbb{Z}_q^{m \times n}$ is chosen uniformly, $\vec{e} \leftarrow \chi^m$ and $\vec{b} = A \cdot \vec{s} - \vec{e} \in \mathbb{Z}_q^m$.

In this paper, we mainly consider an LWE variant that uses ternary secrets and errors $\vec{s}, \vec{e} \in \mathcal{T}^n$, which is exploited in the efficient NTRU-type cryptosystems [26,16,34,40]. We also assume the number of samples m is equal to n, which is common in the algebraically structured LWE such as Ring-LWE (RLWE) [57] or Module-LWE (MLWE) [20] setting. We remark that our technique can be naturally extended to LWE with ternary secrets using errors $\vec{e}$'s with bounded ℓ_∞-norms.

2.4 May's Meet-LWE Attack

The Meet-LWE attack [59] is the state-of-the-art MitM attack proposed by Alexander May to recover the secret key $\vec{s}$ of the LWE instance with ternary secrets and errors. In this section, we review May's classical Meet-LWE attack for further discussion.

Let $\vec{s} \in \mathcal{T}^n(w/2)$ be a ternary weight-w vector and $w^{(0)} := w/2$. Given LWE instance $(A, \vec{b} = A\vec{s} - e \mod q) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$, we first split $\vec{s}$ into two ternary vectors $\vec{s}_1, \vec{s}_2 \in \mathcal{T}^n(w^{(1)})$, where $w^{(1)} \geq w^{(0)}/2$ using Howgrave-Graham's representation technique [46].

Then the LWE equation $A\vec{s} = \vec{b} + \vec{e} \mod q$ can be rewritten as

$$A\vec{s}_1 + \vec{e}_1 = \vec{b} - A\vec{s}_2 + \vec{e}_2 \mod q, \tag{1}$$

where $\vec{e}_1 \in \mathcal{T}^{n/2} \times 0^{n/2}$ and $\vec{e}_2 \in 0^{n/2} \times \mathcal{T}^{n/2}$ such that $\vec{e} = \vec{e}_2 - \vec{e}_1$.

Afterward, let $R^{(1)}$ be the number of representations to represent $\vec{s} = \vec{s}_1 + \vec{s}_1$. We define $\pi_r : \mathbb{Z}_q^n \to \mathbb{Z}_q^r$ as the projection onto the first $r = \lfloor \log_q(R^{(1)}) \rfloor$ coordinates, i.e.,

$$\pi_r : \mathbb{Z}_q^n \to \mathbb{Z}_q^r, \vec{x} = (x_1, ..., x_n) \mapsto (x_1, ..., x_r)$$

Since the codomain of π_r has size $q^r < q^{log_q R^{(1)}} = R^{(1)}$, for a fixed randomly chosen target vector $\vec{t} \in \mathbb{Z}_q^r$, one can expect at least one representation $(\vec{s}_1, \vec{s}_2)$ of $\vec{s}$ satisfying

$$\pi_r(A\vec{s}_1 + \vec{e}_1) = \pi_r(\vec{b} - A\vec{s}_2 + \vec{e}_2) = \vec{t} \mod q$$

exists. We can obtain $\vec{s}_1, \vec{s}_2 \in \mathcal{T}^n$ satisfying the equation (1) by exactly guessing r coordinates of $\pi_r(\vec{e})$ and matching on the remaining $n-r$ coordinates approximately. For the approximate match, it utilizes Odlyzko's locality sensitive hash function, constructing L_1 and L_2 for all candidates of $\vec{s}_1$ and $\vec{s}_2$ and their hash values using the binary tree-based list constructions.

Let $L^{(i)}$ be the level-i list and denote binary tree depth as d. Solving the LWE problem requires constructing the level-0 list in Meet-LWE. For $i \in \{0, 1, .., d-1\}$ and $k \in \{1, 2, .., 2^d\}$, to construct each of level-i list $L_k^{(i)}$, two level-$(i+1)$ lists $L_{2k-1}^{(i+1)}, L_{2k}^{(i+1)}$ are required. In the following, we describe the case of constructing a depth-3 binary tree as an example. First, we define level-1 lists as follows.

$$L_1^{(1)} = \{(\vec{s}_1^{(1)} \in \mathcal{T}^n(w^{(1)}), \ell(A\vec{s}_1^{(1)})) \mid \pi_{r^{(1)}}(A\vec{s}_1^{(1)} + \vec{e}_1) = \vec{t} \mod q\}$$
$$L_2^{(1)} = \{(\vec{s}_2^{(1)} \in \mathcal{T}^n(w^{(1)}), \ell(\vec{b} - A\vec{s}_2^{(1)})) \mid \pi_{r^{(1)}}(\vec{b} - A\vec{s}_2^{(1)} + \vec{e}_2) = \vec{t} \mod q\}$$

Both $L_1^{(1)}$ and $L_2^{(1)}$ have size of $L^{(1)} = S^{(1)}/q^r \approx S^{(1)}/R^{(1)}$, where $S^{(1)}$ is the size of the search space $\mathcal{T}^n(w^{(1)})$. Analogously, we define $\vec{s}_1^{(2)}, \vec{s}_2^{(2)}, \vec{s}_3^{(2)}, \vec{s}_4^{(2)} \in \mathcal{T}^n(w^{(2)})$ such that $\vec{s}_1^{(1)} = \vec{s}_1^{(2)} + \vec{s}_2^{(2)}, \vec{s}_2^{(1)} = \vec{s}_3^{(2)} + \vec{s}_4^{(2)}$ and $w^{(2)} \geq w^{(1)}/2$, obtaining

$$A(\vec{s}_1^{(2)} + \vec{s}_2^{(2)}) + \vec{e}_1 = \vec{b} - A(\vec{s}_3^{(2)} + \vec{s}_4^{(2)}) + \vec{e}_2 \mod q. \tag{2}$$

From the equation (2), all four level-2 lists are defined as

$$L_1^{(2)} = \{(\vec{s}_1^{(2)}, A\vec{s}_1^{(2)})) \mid \pi_{r^{(2)}}(A\vec{s}_1^{(2)}) = \vec{t}' \mod q\},$$
$$L_2^{(2)} = \{(\vec{s}_2^{(2)}, A\vec{s}_2^{(2)}) \mid \pi_{r^{(2)}}(A\vec{s}_2^{(2)} + \vec{e}_1^{(2)}) = \pi_{r^{(2)}}(\vec{t}) - \vec{t}' \mod q\},$$
$$L_3^{(2)} = \{(\vec{s}_3^{(2)}, \vec{b} - A\vec{s}_3^{(2)}) \mid \pi_{r^{(2)}}(\vec{b} - A\vec{s}_3^{(2)}) = \vec{t}'' \mod q\},$$
$$L_4^{(2)} = \{(\vec{s}_4^{(2)}, -A\vec{s}_4^{(2)}) \mid \pi_{r^{(2)}}(-A\vec{s}_4^{(2)} + \vec{e}_2^{(2)}) = \pi_{r^{(2)}}(\vec{t}) - \vec{t}'' \mod q\},$$

where $\vec{t}'$ and $\vec{t}''$ are random target vectors, and the number of guessing coordinates on level-2 is $r^{(2)} = \lfloor \log_q(R^{(2)}) \rfloor$, which is a subset of the $r^{(1)}$ fixed coordinates. On level-3, we enumerate $\vec{s}_i^{(2)}$ by calculating a sum of

$$\vec{s}_{2i-1}^{(3)} \in \mathcal{T}^{\frac{n}{2}}(w^{(3)}) \times 0^{\frac{n}{2}} \text{ and } \vec{s}_{2i}^{(3)} \in 0^{\frac{n}{2}} \times \mathcal{T}^{\frac{n}{2}}(w^{(3)}),$$

where $1 \leq i \leq 4$ and $w^{(3)} \geq w^{(2)}/2$. Once we construct $L_1^{(1)}, L_2^{(1)}$ recursively from the higher level lists, we can find a pair of $(\vec{s}_1, \vec{s}_2)$ by matching the Odlyzko's locality sensitive hash function values and check if $\vec{s} := \vec{s}_1 + \vec{s}_2 \in \mathcal{T}^n(w/2)$, which results in getting the final solution $\vec{s}$. The high-level algorithm of the Meet-LWE attack is elaborated in Algorithm 1.

Algorithm 1: LWE Key Search with Meet-LWE (High-Level) [59]

Require: $(A, \vec{b}) \in \mathbb{Z}_q^{n \times n} \times \mathbb{Z}_q^n$, weight $w \in \mathbb{N}$
Ensure: ternary $\vec{s}$ of Hamming weight w such that $\vec{e} = A\vec{s} - \vec{b}$ *mod* $q \in \mathcal{T}^n$
Represent $\vec{s} = \vec{s}_1 + \vec{s}_2$ choosing one representation strategies among Rep-0, Rep-1, Rep-2 described in [59] for $\vec{s}_1, \vec{s}_2 \in \mathcal{T}^n(w/4)$.
Let $R^{(1)}$ be the number of representations $\vec{s} = \vec{s}_1 + \vec{s}_2$ and $r := \lfloor \frac{1}{2} \log_q(R^{(1)}) \rfloor$.
for all $\pi_r(\vec{e}_1) \in \mathcal{T}^{r/2} \times 0^{r/2}$ and $\pi_r(\vec{e}_2) \in 0^{r/2} \times \mathcal{T}^{r/2}$ **do**
    Construct
    $L_1^{(1)} = \{(\vec{s}_1, \ell(A\vec{s}_1)) |\ \pi_r(A\vec{s}_1 + \vec{e}_1) = \vec{t} \mod q\}$,
    $L_2^{(1)} = \{(\vec{s}_2, \ell(\vec{b} - A\vec{s}_2)) |\ \pi_r(\vec{b} - A\vec{s}_2 + \vec{e}_2) = \vec{t} \mod q\}$ from Eq. (1), using binary tree-based list construction.
for all matches of $(\vec{s}_1, \vec{l}_1), (\vec{s}_2, \vec{l}_2)$ in $L_1 \times L_2$ **s.t.** $\vec{l}_1 = \vec{l}_2$ **do**
    if $((\vec{s} := \vec{s}_1 + \vec{s}_2 \in \mathcal{T}^n(w/2))$ and $(A\vec{s} - \vec{b} \mod q \in \mathcal{T}^n))$ **then**
        return $\vec{s}$

3 Extended Meet-LWE Attack with Ternary Trees

3.1 Our Algorithm in High-Level

In this subsection, we present a high-level overview of the improved Meet-LWE attack. We first split the LWE secret $\vec{s} \in \mathcal{T}^n(w/2)$ into a sum of three n-dimensional vectors $\vec{s} = \vec{s}_1 + \vec{s}_2 + \vec{s}_3$ for $\vec{s}_1, \vec{s}_2, \vec{s}_3 \in \mathcal{T}^n(w/6)$, which results in numerous representations. As in the Meet-LWE attack, we apply (and extend) various strategies to represent $\vec{s}$ named Rep-0, Rep-1-0, and Rep-1-1 as summarized in Table 2. We describe the Rep-0 strategy in Section 4, in which $\vec{s}_1, \vec{s}_2, \vec{s}_3$ are ternary vectors with the Hamming weight $w/3$. Also, Rep-1-0 (Rep-1-1) strategy is dealt in Section 5, in which $\vec{s}_1, \vec{s}_2, \vec{s}_3$ have the Hamming weights larger than $w/3$. In our analysis, we observe that the attack complexity for Rep-2 that represents $\vec{s}$ with $\vec{s}_1, \vec{s}_2, \vec{s}_3 \in \{0, \pm 1, \pm 2\}^n$ is no better than those for Rep-1-0 and Rep-1-1. Hence, we do not address the Rep-2 strategy.

Table 2: Representation Strategies

Represents	-1	0	1
Rep-0	(-1)+0+0 0+(-1)+0 0+0+(-1)	0+0+0	1+0+0 0+1+0 0+0+1
Rep-1-0	(-1)+0+0 0+(-1)+0 0+0+(-1)	0+1+(-1) 1+(-1)+0 0+(-1)+1 (-1)+0+1 1+0+(-1) (-1)+1+0	1+0+0 0+1+0 0+0+1
Rep-1-1	(-1)+0+0 (-1)+1+(-1) 0+(-1)+0 (-1)+(-1)+1 0+0+(-1) 1+(-1)+(-1)	0+1+(-1) 1+(-1)+0 0+(-1)+1 (-1)+0+1 1+0+(-1) (-1)+1+0	1+0+0 1+1+(-1) 0+1+0 (-1)+1+1 0+0+1 1+(-1)+1

By defining $\vec{s} = \vec{s}_1 + \vec{s}_2 + \vec{s}_3$, the LWE equation is rewritten as follows.

$$(A\vec{s}_1 + \vec{e}_1) + (A\vec{s}_2 + \vec{e}_2) = \vec{b} - A\vec{s}_3 + \vec{e}_3 \mod q, \tag{3}$$

where $\vec{e}_1 \in \mathcal{T}^{n/3} \times 0^{2n/3}, \vec{e}_2 \in 0^{n/3} \times \mathcal{T}^{n/3} \times 0^{n/3}$, and $\vec{e}_3 \in 0^{2n/3} \times \mathcal{T}^{n/3}$ such that $\vec{e} = \vec{e}_3 - \vec{e}_2 - \vec{e}_1$.

For a pre-fixed parameter $r > 0$, we apply the projection map $\pi_r : \mathbb{Z}_q^n \to \mathbb{Z}_q^r$ defined by $\pi_r(x_1, \cdots, x_n) = (x_1, \cdots, x_r)$ to the equation (3), achieving

$$\pi_r(A\vec{s}_1 + \vec{e}_1) + \pi_r(A\vec{s}_2 + \vec{e}_2) = \pi_r(\vec{b} - A\vec{s}_3 + \vec{e}_3) \mod q.$$

Suppose that q^r, the size of the range of π_r, is smaller than $\sqrt{R^{(1)}}$, where $R^{(1)}$ is the number of representations. Then, since $q^{2r} < R^{(1)}$, we can expect at least one representation of the solution matches the random target $(\vec{t}_1, \vec{t}_2) \in (\mathbb{Z}_q^r)^2$ satisfying $\pi_r(A\vec{s}_1 + \vec{e}_1) = \vec{t}_1$ and $\pi_r(A\vec{s}_2 + \vec{e}_2) = \vec{t}_2$. Note that this implies $\pi_r(\vec{b} - A\vec{s}_3 + \vec{e}_3) = \vec{t}_1 + \vec{t}_2$.

Hence, we exhaustively search $\pi_r(\vec{e}_i)$'s to construct lists $L_1^{(1)}$, $L_2^{(1)}$, and $L_3^{(1)}$ defined as follows using the tree-based construction:

$$\begin{aligned}
L_1^{(1)} &= \{(\vec{s}_1, \ell(A\vec{s}_1)) \mid \pi_r(A\vec{s}_1 + \vec{e}_1) = \vec{t}_1 \mod q\}, \\
L_2^{(1)} &= \{(\vec{s}_2, \ell(A\vec{s}_2)) \mid \pi_r(A\vec{s}_2 + \vec{e}_2) = \vec{t}_2 \mod q\}, \\
L_3^{(1)} &= \{(\vec{s}_3, \ell(\vec{b} - A\vec{s}_3)) \mid \pi_r(\vec{b} - A\vec{s}_3 + \vec{e}_3) = \vec{t}_1 + \vec{t}_2 \mod q\}.
\end{aligned}$$

For the tree-based list construction, we utilize the ternary trees instead of binary trees except for the top level. With the ternary trees, the number of representations increases drastically from the bottom level, while the Hamming weights of the secrets decrease quickly at a higher level. This leads to reduced complexity compared to the Meet-LWE attack for both time and memory consumption. We describe our algorithm at a high level in Algorithm 2.

Algorithm 2: LWE Key Search with Extended Meet-LWE (High-Level)

1 **Require:** $(A, \vec{b}) \in \mathbb{Z}_q^{n \times n} \times \mathbb{Z}_q^n$, weight $w \in \mathbb{N}$
2 **Ensure:** ternary weight-w $\vec{s}$ satisfying $\vec{e} = A\vec{s} - \vec{b}\ mod\ q \in \mathcal{T}^n$
3 We represent $\vec{s} = \vec{s}_1 + \vec{s}_2 + \vec{s}_3$ using different representation strategies described in Section 4 to 5 for $\vec{s}_1, \vec{s}_2, \vec{s}_3 \in \mathcal{T}^n(w/6)$.
4 Let $R^{(1)}$ be the resulting number of representations. Let $r = \lfloor \frac{1}{2} \log_q(R^{(1)}) \rfloor$
5 **for** all $\pi_r(\vec{e}_1) \in \mathcal{T}^{r/3} \times 0^{r/3} \times 0^{r/3}$ **do**
6 Construct $L_1^{(1)} = \{(\vec{s}_1, \ell(A\vec{s}_1)) |\ \pi_r(A\vec{s}_1 + \vec{e}_1) = \vec{t}_1 \mod q\}$ from Eq. (3), using tree-based list construction
7 **for** all $\pi_r(\vec{e}_2) \in 0^{r/3} \times \mathcal{T}^{r/3} \times 0^{r/3}$ **do**
8 Construct $L_2^{(1)} = \{(\vec{s}_2, \ell(A\vec{s}_2)) |\ \pi_r(A\vec{s}_2 + \vec{e}_2) = \vec{t}_2 \mod q\}$ from Eq. (3), using tree-based list construction
9 **for** all $\pi_r(\vec{e}_3) \in 0^{r/3} \times 0^{r/3} \times \mathcal{T}^{r/3}$ **do**
10 Construct $L_3^{(1)} = \{(\vec{s}_3, \ell(\vec{b} - A\vec{s}_3)) |\ \pi_r(\vec{b} - A\vec{s}_3 + \vec{e}_3) = \vec{t}_1 + \vec{t}_2 \mod q\}$ from Eq. (3), using tree-based list construction
11 **for** all matches of $(\vec{s}_1, \vec{l}_1), (\vec{s}_2, \vec{l}_2), (\vec{s}_3, \vec{l}_3)$ in $L_1 \times L_2 \times L_3$ **s.t.** $\vec{l}_1 \oplus \vec{l}_2 = \vec{l}_3$ **do**
12 **if** $((\vec{s} := \vec{s}_1 + \vec{s}_2 + \vec{s}_3 \in \mathcal{T}^n(w/2))$ and $(A\vec{s} - \vec{b} \mod q \in \mathcal{T}^n))$ **then**
13 return $\vec{s}$

3.2 Correctness

Following the Algorithm 2, we can find a collision $\ell(A\vec{s}_1) \oplus \ell(A\vec{s}_2) = \ell(\vec{b} - A\vec{s}_3)$ by comparing the hash values of candidates where $(\vec{s}_1, \vec{s}_2, \vec{s}_3)$ satisfies the equation (3) on r coordinates. Furthermore, we verify the correctness of our algorithm using the so-called *Match-and-filter* [13,15] in line 12 of Algorithm 2, the process of checking the consistency of solutions. It ensures that the found $\vec{s}$ is a ternary vector with Hamming weight w and $\vec{e}$ is a ternary vector, respectively.

3.3 Attack Complexity

For the first three outer **for** loops in Algorithm 2, we guess $r/3$ coordinates of $\vec{e}$. We denote the complexity of the guessing part of the outer **for** loop as T_{guess} and the list construction part of the inner loop as T_{list}. The time complexity of Algorithm 2 is then $T = T_{\mathsf{guess}} \cdot T_{\mathsf{list}}$.

The list construction complexity is $T_{\mathsf{list}} = 2^{\mathcal{O}(n)}$, and we can compute

$$T_{\mathsf{guess}} = 3^{r/3} \leq 3^{\frac{1}{6} \cdot \log_q R^{(1)}} = 2^{\frac{1}{6}(\log_2 3 / \log_2 q) \cdot \log_2 R^{(1)}}.$$

Since $q = \Omega(n)$ and $\log_2 R^{(1)} = O(n)$, it follows that $T_{\mathsf{guess}} = 2^{\mathcal{O}(\frac{n}{\log n})}$. Hence, the asymptotic complexity is determined by T_{list} as in the original Meet-LWE attack. T_{list} is calculated concretely depending on the different strategies of the representation of $\vec{s}$, which will be computed in Section 4 and 5.

4 Rep-0

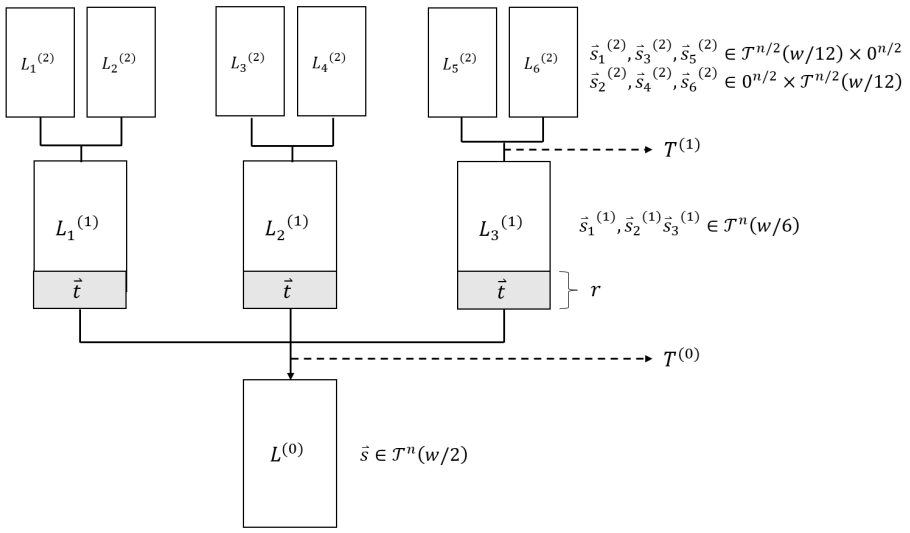

Fig. 1: An Instantiation of Our Algorithm Equipped with REP-0 (depth 2)

For the ternary $\vec{s} \in \mathcal{T}^n(w/2)$ with Hamming weight w, we split $\vec{s}$ into three vectors $\vec{s}_1^{(1)}, \vec{s}_2^{(1)}, \vec{s}_3^{(1)} \in \mathcal{T}^n(w/6)$ where (± 1)-coordinates of $\vec{s}$ are represented as $(\pm 1)+0+0$ or $0+(\pm 1)+0$ or $0+0+(\pm 1)$. In this case, the search space size for each $\vec{s}_i^{(1)}$ is $S^{(1)} = \binom{n}{\frac{w}{6}, \frac{w}{6}, \cdot} \approx 2^{H(\frac{w}{6}\,\frac{w}{6},\cdot)n}$ and the number of representation $R^{(1)}$ is $R^{(1)} = \binom{\frac{w}{2}}{\frac{w}{6}, \frac{w}{6}, \frac{w}{6}}^2 \approx 3^{\omega n}$, where $w = \omega n$.

We construct $L_1^{(1)}, L_2^{(1)}$ and $L_3^{(1)}$ satisfying the equation (3) as follows.

$$\begin{aligned} L_1^{(1)} &= \{(\vec{s}_1, \ell(A\vec{s}_1))|\ \pi_r(A\vec{s}_1 + \vec{e}_1) = \vec{t}_1 \mod q\}, \\ L_2^{(1)} &= \{(\vec{s}_2, \ell(A\vec{s}_2))|\ \pi_r(A\vec{s}_2 + \vec{e}_2) = \vec{t}_2 \mod q\}, \\ L_3^{(1)} &= \{(\vec{s}_3, \ell(b - A\vec{s}_3))|\ \pi_r(\vec{b} - A\vec{s}_3 + \vec{e}_3) = \vec{t}_1 + \vec{t}_2 \mod q\} \end{aligned} \tag{4}$$

The list $L_i^{(1)}$ is of size

$$L_i^{(1)} = \frac{S^{(1)}}{q^r} \approx \frac{S^{(1)}}{\sqrt{R^{(1)}}} = \binom{n}{\frac{w}{6}, \frac{w}{6}, \cdot} \binom{\frac{w}{2}}{\frac{w}{6}, \frac{w}{6}, \frac{w}{6}}^{-1} \approx 2^{\left(H(\frac{w}{6}\,\frac{w}{6},\cdot) - (\frac{\log_2 3}{2})\cdot\omega\right)n}.$$

On the top level, which we set level-2 as in Figure 1, we split $s_1^{(1)} \in \mathcal{T}^n(w/6)$ into $s_1^{(2)} \in \mathcal{T}^{\frac{n}{2}}(w/12) \times 0^{\frac{n}{2}}$ and $s_2^{(2)} \in 0^{\frac{n}{2}} \times \mathcal{T}^{\frac{n}{2}}(w/12)$ using Odlyzko's MitM algorithm, and $s_2^{(1)}$ into $\vec{s}_3^{(2)}, \vec{s}_4^{(2)}$ and $s_3^{(1)}$ into $\vec{s}_5^{(2)}, \vec{s}_6^{(2)}$ in the same manner. The

search space of $s_i^{(2)}$ is of size $S^{(2)} = \binom{\frac{n}{2}}{\frac{w}{12}, \frac{w}{12}, \cdot} \approx 2^{\frac{1}{2}H(\frac{\omega}{6}, \frac{\omega}{6}, \cdot)}$. Also, the size of all six lists $L_1^{(2)}, ..., L_6^{(2)}$ obtained at the top level is the same as that of the search space $L^{(2)} = S^{(2)} \approx 2^{\frac{1}{2}H(\frac{\omega}{6}, \frac{\omega}{6}, \cdot)}$.

Let $T^{(i)}$ be the time complexity to construct each list on level-i by mapping the lists on level-$(i-1)$. Hence, the time complexity $T^{(1)}$ is $T^{(1)} = max\{L^{(2)}, L^{(1)}\}$. Considering the approximate matching on $n-r$ coordinates with Odlyzko's hash function, the time $T^{(2)}$ is $T^{(2)} = max\{L^{(1)}, 2^{-(n-r)} \cdot (L^{(1)})^2\}$, which results in $L^{(1)}$.

Hence, we can compute the total run time of list construction as $T_{\mathsf{list}} = max\{T^{(1)}, T^{(0)}\} = max\{L^{(2)}, L^{(1)}\}$. The log complexity of list construction can also be represented as follows.

$$\log_2 T_{\mathsf{list}} = max\left\{\frac{1}{2}H\left(\frac{\omega}{6}, \frac{\omega}{6}, \cdot\right), \left(H\left(\frac{\omega}{6}, \frac{\omega}{6}, \cdot\right) - \left(\frac{\log_2 3}{2}\right) \cdot \omega\right) \cdot n\right\}.$$

Let T_{guess} be the time complexity to guess each $r/3$ coordinates of random ternary vector $\vec{e}_1, \vec{e}_2, \vec{e}_3$, where $r = \lfloor \frac{1}{2} \log_q(R^{(1)}) \rfloor$ is the total number of guessing components of $\vec{e}$. The guessing complexity can be computed as

$$T_{\mathsf{guess}} = 3^{r/3} \leq 2^{\frac{1}{6}(\log_2 3/\log_2 q) \cdot \log_2 R^{(1)}}.$$

The total complexity is $T = T_{\mathsf{list}} \cdot T_{\mathsf{guess}}$.

5 REP-1

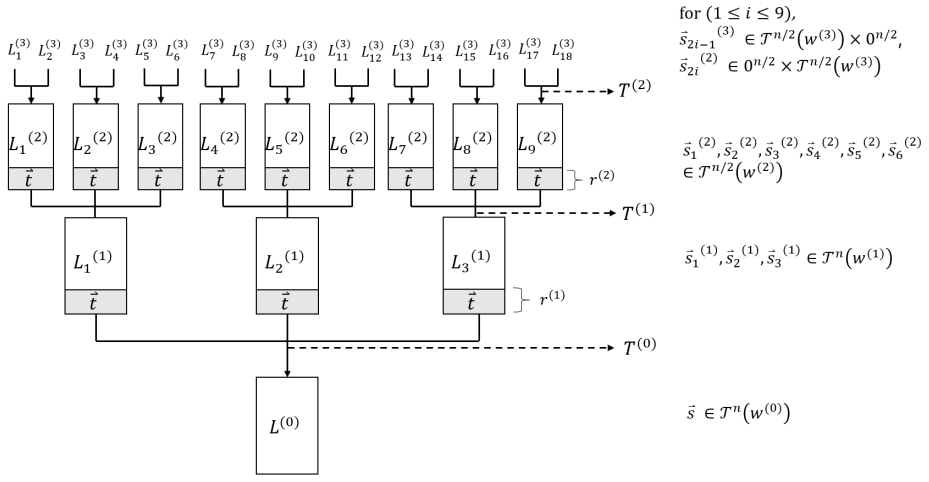

Fig. 2: An Instantiation of Our Algorithm Equipped with REP-1 (when the tree depth is $d = 3$)

Table 3: Number of cases of different level-j representations for the coordinates of $\vec{s}_{3k-2}^{(j)}$, $\vec{s}_{3k-1}^{(j)}$, and $\vec{s}_{3k-1}^{(j)}$, where $\vec{s}_{3k-2}^{(j)} + \vec{s}_{3k-1}^{(j)} + \vec{s}_{3k}^{(j)} = \vec{s}_{k}^{(j-1)}$. The number of each case is denoted by $\epsilon_{10}^{(j)}$ or $\epsilon_{11}^{(j)}$ in column '# of Cases'.

# of Cases	$\vec{s}_{3k-2}^{(j)}$	$\vec{s}_{3k-1}^{(j)}$	$\vec{s}_{3k}^{(j)}$	$\vec{s}_{k}^{(j-1)}$	# of Cases	$\vec{s}_{3k-2}^{(j)}$	$\vec{s}_{3k-1}^{(j)}$	$\vec{s}_{3k}^{(j)}$	$\vec{s}_{k}^{(j-1)}$
$\epsilon_{10}^{(j)}$	1	-1	0	0	$\epsilon_{11}^{(j)}$	1	-1	1	1
$\epsilon_{10}^{(j)}$	1	0	-1	0	$\epsilon_{11}^{(j)}$	-1	1	1	1
$\epsilon_{10}^{(j)}$	0	1	-1	0	$\epsilon_{11}^{(j)}$	1	1	-1	1
$\epsilon_{10}^{(j)}$	-1	1	0	0	$\epsilon_{11}^{(j)}$	1	-1	-1	-1
$\epsilon_{10}^{(j)}$	-1	0	1	0	$\epsilon_{11}^{(j)}$	-1	1	-1	-1
$\epsilon_{10}^{(j)}$	0	-1	1	0	$\epsilon_{11}^{(j)}$	-1	-1	1	-1

In Section 4, Rep-0 represents (± 1)-coordinates of $\vec{s}$ as $(\pm 1)+0+0$, $0+(\pm 1)+0$ and $0+0+(\pm 1)$, but it represents 0-coordinates as $0 = 0+0+0$ only. In Rep-1, we consider additional representations of (± 1)- and 0-coordinates of $\vec{s}$, which are elaborated in Table 3. Here, we denote additional optimization parameters $\epsilon_{10}^{(j)}$ and $\epsilon_{11}^{(j)}$ as the number of each representation. For a total of six representations of 0-coordinate, we assume each representation occurs $\epsilon_{10}^{(j)}$ times on level j. For example, 0 is represented as $1+(-1)+0$ for $\epsilon_{10}^{(j)}$ 0-coordinates on level j. Also, we assume, for each (additional) representation of ± 1 in Table 3, it occurs $\epsilon_{11}^{(j)}$ times. The Rep-1 strategy is exploited in the original Meet-LWE attack as well; however, our attack considers an additional optimization parameter because of the new types of representations. We parse REP-1 into two representation strategies: when $\epsilon_{11}^{(j)} = 0$, we call it REP-1-0, and otherwise, we call it REP-1-1. Hence, we present the analysis for REP-1-1, and one can simply substitute $\epsilon_{11}^{(j)}$'s with 0 to achieve those for REP-1-0. We describe how to construct the level-j lists in the following.

On level-1, we split $\vec{s} \in \mathcal{T}^n(w^{(0)})$ into $\vec{s}_1^{(1)}, \vec{s}_2^{(1)}, \vec{s}_3^{(1)} \in \mathcal{T}^n(w^{(1)})$ using the REP-1 representation strategy. The Hamming weight $w^{(1)}$ of $\vec{s}_i^{(1)}$ is calculated as

$$w^{(1)} = \frac{w^{(0)} - 3 \cdot \epsilon_{11}^{(1)}}{3} + 2 \cdot \epsilon_{10}^{(1)} + 3 \cdot \epsilon_{11}^{(1)} = \frac{w^{(0)}}{3} + 2 \cdot \epsilon_{10}^{(1)} + 2 \cdot \epsilon_{11}^{(1)},$$

since we alternate $3 \cdot \epsilon_{11}^{(1)}$ ± 1's (*resp.*, $6 \cdot \epsilon_{10}^{(1)}$ 0's) in $\vec{s}$ obtaining $3 \cdot \epsilon_{11}^{(1)}$ ± 1's (*resp.*, $2 \cdot \epsilon_{10}^{(1)}$ ± 1's) in each $\vec{s}_i^{(1)}$. Also, $R^{(1)}$ and $S^{(1)}$ can be computed as follows.

$$R^{(1)} = \binom{w^{(0)}}{\epsilon_{11}^{(1)}, \epsilon_{11}^{(1)}, \epsilon_{11}^{(1)}, \frac{w^{(0)}}{3} - \epsilon_{11}^{(1)}, \frac{w^{(0)}}{3} - \epsilon_{11}^{(1)}, \frac{w^{(0)}}{3} - \epsilon_{11}^{(1)}}^{2} \cdot \binom{n - 2w^{(0)}}{\epsilon_{10}^{(1)}, \epsilon_{10}^{(1)}, \epsilon_{10}^{(1)}, \epsilon_{10}^{(1)}, \epsilon_{10}^{(1)}, \epsilon_{10}^{(1)}, .},$$

$$S^{(1)} = \binom{n}{w^{(1)}, w^{(1)}, .},$$

where the second factor in $R^{(1)}$ is for the additional case of representations to denote 0-components. We remark that both the number of representations $R^{(1)}$ and the search space size $S^{(1)}$ are increased compared to those of Rep-0 due to the additional representations.

For the level-1 lists, we first choose random target vectors $\vec{t}_1, \vec{t}_2 \in \mathbb{Z}_q^{r^{(1)}}$, and define $L_1^{(1)}, L_2^{(1)}, L_3^{(1)}$ of size $L^{(1)} \approx S^{(1)}/\sqrt{R^{(1)}}$ as

$$
\begin{aligned}
L_1^{(1)} &= \{(\vec{s}_1^{(1)} \in \mathcal{T}^n(w^{(1)}), \ell(A\vec{s}_1^{(1)})) \mid \pi_{r^{(1)}}(A\vec{s}_1^{(1)} + \vec{e}_1) = \vec{t}_1 \mod q\},\\
L_2^{(1)} &= \{(\vec{s}_2^{(1)} \in \mathcal{T}^n(w^{(1)}), \ell(A\vec{s}_2^{(1)})) \mid \pi_{r^{(1)}}(A\vec{s}_2^{(1)} + \vec{e}_2) = \vec{t}_2 \mod q\},\\
L_3^{(1)} &= \{(\vec{s}_3^{(1)} \in \mathcal{T}^n(w^{(1)}), \ell(\vec{b} - A\vec{s}_3^{(1)})) \mid \pi_{r^{(1)}}(\vec{b} - A\vec{s}_3^{(1)} + \vec{e}_3) = \vec{t}_1 + \vec{t}_2 \mod q\}.
\end{aligned}
$$

On level $2 \le j < d$, we construct level-j lists $L_1^{(j)}, \cdots, L_{3j}^{(j)}$ with $\vec{s}_1^{(j)}, \cdots, \vec{s}_{3j}^{(j)}$ of weight

$$w^{(j)} = \frac{w^{(j-1)}}{3} + 2 \cdot \epsilon_{10}^{(j)} + 2 \cdot \epsilon_{11}^{(j)}.$$

Also, $R^{(j)}$ and $S^{(j)}$ on level-j can be obtained as follows.

$$
R^{(j)} = \binom{w^{(j-1)}}{\epsilon_{11}^{(j)}, \epsilon_{11}^{(j)}, \epsilon_{11}^{(j)}, \frac{w^{(j-1)}}{3} - \epsilon_{11}^{(j)}, \frac{w^{(j-1)}}{3} - \epsilon_{11}^{(j)}, \frac{w^{(j-1)}}{3} - \epsilon_{11}^{(j)}}^2 \cdot \binom{n - 2w^{(j-1)}}{\epsilon_{10}^{(j)}, \epsilon_{10}^{(j)}, \epsilon_{10}^{(j)}, \epsilon_{10}^{(j)}, \epsilon_{10}^{(j)}, \epsilon_{10}^{(j)}, .},
$$

$$
S^{(j)} = \binom{n}{w^{(j)}, w^{(j)}, .}.
$$

The list size $L^{(j)}$ on the j-th level is estimated as $L^{(j)} \approx S^{(j)}/\sqrt{R^{(j)}}$.

In the following, we provide a detailed description of level-2 lists while omitting the level-j lists for $2 < j < d$, which are defined in a similar manner. On level-2, we split $\vec{s}_1^{(1)}$ into $\vec{s}_1^{(2)} + \vec{s}_2^{(2)} + \vec{s}_3^{(2)}$ and do the same for $\vec{s}_2^{(1)}$ and $\vec{s}_3^{(1)}$, using the REP-1 representation strategy. After choosing six random target vectors $\vec{t}_1, \vec{t}_2, \vec{t}_1', \vec{t}_2', \vec{t}_1'', \vec{t}_2'' \in \mathbb{Z}_q^{r^{(2)}}$, we construct the following level-2 lists according to the equation (3).

$$
\begin{aligned}
L_1^{(2)} &= \{(\vec{s}_1^{(2)}, A\vec{s}_1^{(2)})) \mid \pi_{r^{(2)}}(A\vec{s}_1^{(2)}) = \vec{t}_1' \mod q\}\\
L_2^{(2)} &= \{(\vec{s}_2^{(2)}, A\vec{s}_2^{(2)}) \mid \pi_{r^{(2)}}(A\vec{s}_2^{(2)} + \vec{e}_1^{(2)}) = \vec{t}_2' \mod q\}\\
L_3^{(2)} &= \{(\vec{s}_3^{(2)}, A\vec{s}_3^{(2)}) \mid \pi_{r^{(2)}}(A\vec{s}_3^{(2)}) = \pi_{r^{(2)}}(\vec{t}_1) - (\vec{t}_1' + \vec{t}_2') \mod q\}\\
L_4^{(2)} &= \{(\vec{s}_4^{(2)}, A\vec{s}_4^{(2)})) \mid \pi_{r^{(2)}}(A\vec{s}_4^{(2)}) = \vec{t}_1'' \mod q\}\\
L_5^{(2)} &= \{(\vec{s}_5^{(2)}, A\vec{s}_5^{(2)}) \mid \pi_{r^{(2)}}(A\vec{s}_5^{(2)} + \vec{e}_2^{(2)}) = \vec{t}_2'' \mod q\}\\
L_6^{(2)} &= \{(\vec{s}_6^{(2)}, A\vec{s}_6^{(2)}) \mid \pi_{r^{(2)}}(A\vec{s}_6^{(2)}) = \pi_{r^{(2)}}(\vec{t}_2) - (\vec{t}_1'' + \vec{t}_2'') \mod q\}\\
L_7^{(2)} &= \{(\vec{s}_7^{(2)}, -A\vec{s}_7^{(2)})) \mid \pi_{r^{(2)}}(-A\vec{s}_7^{(2)}) = \vec{t}_1''' \mod q\}\\
L_8^{(2)} &= \{(\vec{s}_8^{(2)}, -A\vec{s}_8^{(2)}) \mid \pi_{r^{(2)}}(-A\vec{s}_8^{(2)} + \vec{e}_3^{(2)}) = \vec{t}_2''' \mod q\}\\
L_9^{(2)} &= \{(\vec{s}_9^{(2)}, \vec{b} - A\vec{s}_9^{(2)}) \mid \pi_{r^{(2)}}(\vec{b} - A\vec{s}_9^{(2)}) = \pi_{r^{(2)}}(\vec{t}_1 + \vec{t}_2) - (\vec{t}_1''' + \vec{t}_2''') \mod q\}
\end{aligned}
$$

On the top level (level-d), the lists are simply defined as

$$L_{2k-1}^{(d)} = \mathcal{T}^{n/2}(w^{(d)}) \times 0^{n/2}, \text{ and } L_{2k}^{(d)} = 0^{n/2} \times \mathcal{T}^{n/2}(w^{(d)}),$$

for $1 \le k \le 3^{d-1}$, where $w^{(d)} = w^{(d-1)}/2$.

5.1 Attack Complexity

In this section, we compute the time complexity of our algorithm equipped with REP-1, which is dominated by the maximum of the list construction complexities. We recall that the runtime for list construction on i-th level is denoted by $T^{(i)}$. The runtime for the list construction T_{list} and the memory cost M are

$$T_{\mathsf{list}} = max\{T^{(0)}, \cdots, T^{(d)}\}, \quad M = max\{L^{(1)}, \cdots, L^{(d)}\},$$

where $T^{(i)}$'s are computed as follows.

On the level-d (top level), we split $\vec{s}_i^{(j-1)}$ with two $n/2$-dimensional vectors using the splitting manner in Odlyzko's MitM. This gives us the time complexity of the level-d list,

$$T^{(d)} = L^{(d)} = \sqrt{S^{(d-1)}}.$$

The construction of level-$(d-1)$ lists with matching level-d lists on $r^{(d-1)}$ coordinates is size of

$$T^{(d-1)} = \frac{\left(L^{(d)}\right)^2}{q^{r^{(d-1)}}} = \frac{\left(L^{(d)}\right)^2}{\sqrt{R^{(d-1)}}}.$$

In general, for $1 < j < d$, the construction of level-$(j-1)$ lists with matching level-j lists on remaining coordinates, after removing $r^{(j)}$ from $r^{(j-1)}$ coordinates, is as follows.

$$T^{(j-1)} = \frac{\left(L^{(j)}\right)^3}{q^{r^{(j-1)}-r^{(j)}}} = \frac{\left(L^{(j)}\right)^3}{\sqrt{R^{(j-1)}}}.$$

Recursively, we can obtain the level-1 lists $L_1^{(1)}, L_2^{(1)}$ and $L_3^{(1)}$. Finally, we find the solution by matching them approximately with Odlyzko's hash function values. At this time, $r^{(1)}$ coordinates have already been matched. Hence, we construct level-0 lists by matching on $n - r^{(1)}$ coordinates of locality sensitive hash values.

$$T^{(0)} = (2^{n-r^{(1)}})^2 \cdot \left(\frac{L^{(1)}}{2^{n-r^{(1)}}}\right)^3 = \frac{\left(L^{(1)}\right)^3}{2^{n-r^{(1)}}},$$

where the term $2^{n-r^{(1)}}$ represents the number of cases of different outputs (bins) of hash values for $L_1^{(1)}$ and $L_2^{(1)}$, and the term $\left(\frac{L^{(1)}}{2^{n-r^{(1)}}}\right)$ represents the number of solutions in each bin corresponding to $\ell(\vec{s}_1^{(1)})$, $\ell(\vec{s}_2^{(1)})$, and $\ell(\vec{s}_1^{(1)}) \oplus \ell(\vec{s}_2^{(1)})$ taken as an average value for each $L_1^{(1)}$, $L_2^{(1)}$ and $L_3^{(1)}$.

6 Asymptotic and Non-asymptotic Comparison

In this section, we optimize the parameters in our attack with tree depth less than or equal to $d = 4$ since no better result is achieved for $d \geq 5$, and compare the resulting time complexities with those of Meet-LWE. We present the optimized asymptotic and non-asymptotic runtime with comparison in Section 6.1 and 6.2, respectively. We report the runtime estimation for analyzing the KpqC Round 2 candidates in Section 6.3.

6.1 Asymptotic Comparison

As analyzed in Section 3.3, the asymptotic complexity of our algorithm is determined by T_{list}, ignoring the term T_{guess} as in the original Meet-LWE. In Table 4, we summarize the list construction complexity T_{list} for our algorithms for different relative weights ω and compare them with those of May's Meet-LWE. More precisely, since we can determine $T_{\mathsf{list}} := 2^{c(\omega)\cdot n(1+o(1))}$ for some constant $c(\omega)$, we present $c(\omega)$ for each ω in Table 4. For all relative weights $\omega \in \{0.3, 0.375, 0.441, 0.5, 0.62, 0.667\}$, our attack provides a better asymptotic complexity than Meet-LWE as shown in the 6-th and 10-th columns in Table 4.

Table 4: Asymptotics of Our Algorithm Compared to May's Meet-LWE for the Respective Representation Strategies

ω	Odlyz.	May [59]				Ours			
		REP-0	REP-1	REP-2	$\log_{\mathcal{S}} T_{\mathsf{list}}$	REP-0	REP-1-0	REP-1-1	$\log_{\mathcal{S}} T_{\mathsf{list}}$
0.3	0.591	0.469	0.298	0.295	0.25	0.330	0.290	0.247	**0.21**
0.375	0.665	0.523	0.323	0.318	0.24	0.369	0.322	0.286	**0.22**
0.441	0.716	0.561	0.340	0.334	0.23	0.395	0.323	0.293	**0.21**
0.5	0.750	0.588	0.356	0.348	0.23	0.417	0.360	0.315	**0.21**
0.62	0.790	0.625	0.389	0.371	0.24	0.470	0.398	0.340	**0.22**
0.667	0.793	0.634	0.407	0.379	0.24	0.508	0.449	0.351	**0.22**

6.2 Non-asymptotic Comparison

Table 5 shows the optimized time complexities in bits of our algorithm compared to those of Meet-LWE on the REP-0 representations. In Section 3.3, we denote by $\log T = \log T_{\mathsf{list}} + \log T_{\mathsf{guess}}$ the total time complexity in bits where T_{list} is the time complexity of list construction, and T_{guess} is the time complexity of the guessing part. For example, the NTRU-Encrypt (509,2048,254) has $\log T = 231$ total complexity in bits where $\log T_{\mathsf{list}}$ and $\log T_{\mathsf{guess}}$ values are 213 bit and 18 bit, respectively. Compared to Meet-LWE, our REP-0 has approximately 20.38% to 29.15% less complexity.

Recall that in Section 5, two representation methods (REP-1-0, REP-1-1) are used depending on the value of $\epsilon_{11}^{(j)}$. When $\epsilon_{11}^{(j)}$ has a non-zero value, we call it REP-1-1. We present the optimized complexity for REP-1-0 and REP-1-1, respectively, in Table 6. In our algorithm, we optimized every instance with depth-3 search trees since increasing the depth to 4 does not improve the performance.

The 'params' column for REP-1-0 in Table 6 is added to represent the optimized depth of a search tree and the number of additional ± 1's to represent 0 in every level-i list. As an example, the NTRU Prime parameter (653,4621,288) has its params column (3:13,1) for our attack with REP-1-0, which means it optimizes the search trees in list construction of depth 3, where level-1 and 2 lists receive amounts of 13 and 1 additional ± 1's to represent 0, respectively. In 'params' column (3: 8,0,0,2,0,0) for REP-1-1 in Table 6, the first three components correspond to level-j ($j = 1, 2, 3$) as $\epsilon_{10}^{(j)}$

Table 5: Comparison on Non-asymptotic Complexity, May's [59] vs. Ours for REP-0

(n, q, w)	May [bit]	Ours [bit]	Reduced (%)
NTRU-Encrypt			
$(509, 2048, 254)$	$305 = 287+18$	$\mathbf{231} = 213+18$	24.26%
$(677, 2048, 254)$	$364 = 347+18$	$\mathbf{268} = 250+18$	26.37%
$(821, 4096, 510)$	$520 = 487+33$	$\mathbf{414} = 379+34$	20.38%
NTRU Prime			
$(653, 4621, 288)$	$370 = 352+18$	$\mathbf{279} = 260+19$	24.59%
$(761, 4591, 286)$	$408 = 390+18$	$\mathbf{299} = 380+19$	26.72%
$(857, 5167, 322)$	$459 = 439+20$	$\mathbf{337} = 316+21$	26.58%
BLISS I+II			
$(512, 12289, 154)$	$247 = 238+9$	$\mathbf{175} = 167+9$	29.15%
GLP I			
$(512, 8383489, 342)$	$325 = 314+12$	$\mathbf{257} = 246+12$	20.92%

Table 6: Non-asymptotic Attack Complexity of Our Attack Instantiated with REP-1-0 and REP-1-1

(n, q, w)	REP-1-0 [bit]	params	REP-1-1 [bit]	params
NTRU-Encrypt				
$(509, 2048, 254)$	$\mathbf{192} = 173+19$	3: 13,1	$202 = 167+35$	3: 8,0,0,2,0,0
$(677, 2048, 254)$	$\mathbf{214} = 190+24$	3: 19,1	$233 = 196+37$	3: 9,0,0,1,0,0
$(821, 4096, 510)$	$\mathbf{346} = 318+28$	3: 14,1	$351 = 280+71$	3: 19,3,0,10,0,0
NTRU Prime				
$(653, 4621, 288)$	$\mathbf{232} = 213+19$	3: 13,1	$240 = 201+39$	3: 11,1,0,2,0,0
$(761, 4591, 286)$	$\mathbf{242} = 218+24$	3: 21,1	$256 = 206+50$	3: 19,1,0,2,0,0
$(857, 5167, 322)$	$\mathbf{273} = 247+26$	3: 22,1	$291 = 250+41$	3: 11,0,0,1,0,0
BLISS I+II				
$(512, 12289, 154)$	$\mathbf{151} = 136+15$	3: 17,1	$157 = 137+20$	3: 6,0,0,2,0,0
GLP I				
$(512, 8383489, 342)$	$230 = 220+10$	3: 13,4	$\mathbf{217} = 194+23$	3: 9,3,0,10,0,0

and the latter three correspond to level-j ($j = 1, 2, 3$) as $\epsilon_{11}^{(j)}$. We observe that while GLP I showed better complexity compared to REP-1-0, the rest performed better with REP-1-1.

Table 7: Comparison on Non-asymptotic Complexity, May's [59] REP-2 vs. Ours

(n, q, w)	May [bit]	params	Ours [bit]	params	Reduced
NTRU-Encrypt					
$(509, 2048, 254)$	227 = 189+38	4: 26,2,17,3	**192** = 173+19	3: 13,1	15.4%
$(677, 2048, 254)$	273 = 231+42	4: 32,1,15,1	**214** = 190+24	3: 19,1	21.6%
$(821, 4096, 510)$	378 = 318+60	4: 34,5,30,6	**346** = 318+28	3: 14,1	8.5%
NTRU Prime					
$(653, 4621, 288)$	272 = 229+42	4: 36,2,22,5	**232** = 213+19	3: 13,1	14.7%
$(761, 4591, 286)$	301 = 258+43	4: 36,1,17,2	**242** = 218+24	3: 21,1	19.6%
$(857, 5167, 322)$	338 = 291+47	4: 37,2,19,2	**273** = 247+26	3: 22,1	19.2%
BLISS I+II					
$(512, 12289, 154)$	187 = 163+24	4: 27,0,11,1	**151** = 136+15	3: 17,1	19.3%
GLP I					
$(512, 8383489, 342)$	225 = 206+20	4: 22,3,19,4	**217** = 194+23	3: 9,3,0,10,0,0	3.6%

In Table 7, we compare the best non-asymptotic time complexity of our attack to that from [59]. For May's Meet-LWE attack, REP-2 shows the best complexity, and we take the complexity numbers from their paper [59]. We observe that our best time complexity is reduced by 8 to 65-bits, proportionally by 3.6% to 21.6%, compared to the best results from May's Meet-LWE attack.

6.3 Evaluation for KpqC Round 2 Candidates

In this Section, we analyze the security of the ternary LWE-based schemes, a total of 2 candidates (NTRU+, SMAUG-T), among the lattice-based algorithms submitted to KpqC Competition Round 2 [25], using our attack. NTRU+ [1,49] is an algorithm that improves the efficiency of the existing NTRU scheme, following the strategy to construct NTT-friendly settings for NTRU introduced in NTTRU [58] and NTRU-B [37] and introducing a new message encoding to efficiently achieve the negligible worst-case correctness error. The security of NTRU+ relies on the NTRU problem and RLWE with ternary secrets and errors. SMAUG-T [2,29] is an algorithm designed by merging SMAUG and TiGER, which were KpqC Round 1 schemes. It is based on the module lattice problems (MLWE and MLWR) using the sparse ternary secret keys with fixed Hamming weights.

When estimating the security of these two schemes, we use parameter sets (n, q) presented in their specification documents for input parameter settings. For the weight parameter w of s, since the RLWE secret s in NTRU+ follows a centered binomial distribution, we assume $w = n/2$, which is an average value of the Hamming weights. Therefore, the attack complexity for NTRU+ is a rough estimation for lower bound, since the secrets do not have fixed Hamming weights. For SMAUG-T, we select the

minimum value between h_r and h_s as w. h_r and h_s are the hamming weight of a randomness r used for encryption and of a sparse ternary s, respectively.

The security analysis results using our attack on NTRU+ and SMAUG-T are shown in Table 8, Table 9 and Table 10.

Table 8: Security Evaluation of KpqC Round 2 Schemes (NTRU+, SMAUG-T) with Our Attack Instantiated with REP-0

Parameters	(n, q, w)	May [bit]	Ours [bit]	Reduced (%)
NTRU+576	$(576, 3457, 288)$	341 = 323+18	**261** = 242+20	23.46
NTRU+768	$(768, 3457, 384)$	455 = 430+25	**349** = 322+26	23.3
NTRU+864	$(864, 3457, 432)$	513 = 484+29	**392** = 363+30	23.59
NTRU+1152	$(1152, 3457, 576)$	684 = 646+38	**524** = 484+40	23.59
TiMER	$(512, 1024, 100)$	192 = 185+7	**135** = 128+7	29.69
SMAUG-T128	$(512, 1024, 132)$	227 = 217+10	**165** = 155+10	27.31
SMAUG-T192	$(768, 2048, 151)$	289 = 279+10	**208** = 197+11	28.03
SMAUG-T256	$(1280, 2048, 160)$	361 = 351+10	**253** = 241+11	29.92

In Table 8, we computed our attack's complexity with REP-0 and compared them with those of Meet-LWE. For all parameters, the estimated complexities of our attack defeat those of Meet-LWE, ranging from 23.3% to 29.92%.

Table 9: Security Evaluation of KpqC Round 2 Schemes (NTRU+, SMAUG-T) with Our Attack Instantiated with REP-1-0 and REP-1-1

Parameters	REP-1-0 [bit]	params	REP-1-1 [bit]	params
NTRU+576	**221** = 200+21	3: 16,1	230 = 191+39	3: 10,0,0,3,0,0
NTRU+768	**287** = 261+26	3: 18,1	302 = 243+59	3: 18,1,0,5,0,0
NTRU+864	**319** = 288+31	3: 22,1	338 = 272+66	3: 18,2,0,7,0,0
NTRU+1152	**433** = 397+36	3: 20,1	447 = 361+86	3: 22,2,0,9,0,0
TiMER	130 = 115+15	3: 12,1	**122** = 98+24	3: 8,0,0,0,0,0
SMAUG-T128	**147** = 132+16	3: 11,2	149 = 128+21	3: 4,0,0,1,0,0
SMAUG-T192	**182** = 161+21	3: 17,1	183 = 155+28	3: 8,0,0,0,0,0
SMAUG-T256	**231** = 210+21	3: 15,1	238 = 227+11	3: 0,0,0,0,0,0

In Table 9, we compute the time complexity of REP-1-0 and REP-1-1, respectively, and compare the two. We observe that REP-1-0 has better complexity compared to REP-1-1, except for the TiMER parameter.

From Table 10, we can observe that our best time complexity is reduced by 20 bits to 86 bits, proportionally by 11.98% to 18.62%, compared to May's REP-2 which is the best result from the Meet-LWE attack. We show that our algorithm admits improved

Table 10: Security Evaluation of KpqC Round 2 schemes (NTRU+, SMAUG-T) with May's [59] REP-2 vs. Ours

Parameters	May [bit]	params	Ours [bit]	params	Reduced
NTRU+576	263 = 228+36	4: 21,2,9,0	**221** = 200+21	3: 16,1	15.97%
NTRU+768	349 = 302+47	4: 24,3,14,1	**287** = 261+26	3: 18,1	17.77%
NTRU+864	392 = 339+53	4: 29,3,14,3	**319** = 288+31	3: 22,1	18.62%
NTRU+1152	519 = 448+71	4: 35,5,19,3	**433** = 397+36	3: 20,1	16.57%
TiMER	144 = 124+21	4: 14,0,4,0	**122** = 98+24	3: 8,0,0,0,0,0	15.28%
SMAUG-T128	167 = 147+20	4: 10,0,2,0	**147** = 132+16	3: 11,2	11.98%
SMAUG-T192	214 = 192+21	4: 12,0,1,0	**182** = 161+21	3: 17,1	14.95%
SMAUG-T256	283 = 255+29	4: 15,1,3,0	**231** = 210+21	3: 15,1	18.37%

attack cost than May's attack, for NTRU+ and SMAUG-T.

In their specification document [2], SMAUG-T estimates the classical security of parameters with two methods, denoted as 'classical core-SVP' and 'beyond core-SVP'. According to [2], the 'classical core-SVP' estimates the classical security via the lattice estimator using the cost model 'ADPS16' which represents the conservative core-SVP model [10,3]. The 'beyond core-SVP' model estimates the security via lattice estimator without the core-SVP model and Meet-LWE cost estimation. They suggest four parameter sets named TiMER, SMAUG-T128, SMAUG-T192, and SMAUG-T256, for which the estimated time complexities of our attack are 122 bits, 147 bits, 182 bits, and 231 bits, respectively. The claimed security using the 'beyond core-SVP' from [2] were 135.3 bits, 144.7 bits, 202.0 bits, and 274.6 bits, respectively. For TiMER, SMAUG-T192, and SMAUG-T256 parameters, the estimated attack complexities are lower in security by 13.3 bits, 20 bits, and 43.6 bits than claimed.

When comparing our results with the classical security levels claimed in the proposal document with the 'classical core-SVP', which were 120.0 bits, 120.0 bits, 181.7 bits, and 264.5 bits, respectively, we also observe that the SMAUG-T256 parameter has a lower security estimation by 33.5 bits from the claimed one.

7 Applicability to Hybrid Attacks

In recent research, hybrid attacks have been developed by combining lattice reduction algorithms with combinatorial guessing strategies to reduce the overall attack complexities for ternary secret LWE. These attacks fall into two categories: hybrid primal attacks and hybrid dual attacks. This section provides an overview of these two types of attacks and explores the applicability of integrating our enhanced Meet-LWE attack with existing hybrid approaches. However, we consider it a future work to assess the asymptotic and non-asymptotic complexity of specific lattice-based primitives using the hybrid algorithms in conjunction with our improved Meet-LWE attack.

7.1 Hybrid Dual Attacks

At EUROCRYPT 2017, Albrecht introduced a "hybrid dual attack" for analyzing the hardness of the binary or ternary decision-LWE problem [6]. The hybrid dual attack involves two phases: the lattice-reduction phase and the guessing phase. In the first phase, we divide A into two parts $A = (A_1|A_2) \in \mathbb{Z}_q^{m\times r} \times \mathbb{Z}_q^{m\times(n-r)}$ (respectively the ternary $\vec{s} = (\vec{s}_1|\vec{s}_2) \in \{0, \pm 1\}^r \times \{0, \pm 1\}^{n-r}$). Then we can amputate A_2 (respectively $\vec{s}_2$) by the dual attack and generate a new lower-dimensional LWE-like instance. In the second phase, we guess the entries of part $\vec{s}_2$ and detect the distribution of the error vector in the new instance generated from the first phase. Some subsequent works are published in [7,31,38,17].

Although hybrid dual attacks are considered feasible for evaluating the security of current LWE-based or NTRU-based proposals, it was not fully discovered in May's paper. The work in [17] replaces the exhaustive search phase in traditional hybrid dual attacks with the more efficient Meet-LWE algorithm, where the approach was analyzed and tested against LWE instances with FHE-type parameters, comparing its performance to existing hybrid dual attacks. In their paper, the obstacle to constructing a hybrid dual attack with Meet-LWE is that, when reducing the LWE dimension with the dual attack, the Hamming weight of the reduced secret is not known. Hence, they need to exhaustively search for the weight of the secret of the LWE instance with reduced dimension in a possible range to apply the Meet-LWE attack for it. After the exhaustive search for the Hamming weights, the remaining analysis is similar to that in the Meet-LWE attack. Thus, we consider that it is adaptable and possibly more efficient to integrate our improved algorithm with a hybrid provable dual Meet-LWE [17,61]. We leave the detailed analysis and optimization for future work.

7.2 Hybrid Primal Attacks

The concept behind primal attacks involves embedding the given LWE instance $(A, \vec{b}) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$ into a higher-dimensional lattice with basis $B \in \mathbb{Z}_q^{(m+n+1)\times(m+n+1)}$. Here some coefficient vector $\vec{x} \in \mathbb{Z}_q^{m+n+1}$ exists such that $\vec{x}B = (\vec{s}, \vec{e}, 1)$. The LWE problem is then divided into two parts based on a guessing dimension $r \leq n$, with the matrix B and the secret-error vector being split accordingly. For the first part of $(m + n + 1 - r)$-dimensional sub-lattice, we apply a reduction algorithm to facilitate Babai's NP algorithm [12]. Subsequently, we utilize combinatorial guessing strategies to find the r-dimensional partial secret key $\vec{s}$ such that its generated vector can recover $\vec{e}$ by the NP algorithm using the reduced basis. This approach essentially constitutes a lattice decoding attack due to the use of Babai's NP algorithm. Related works can be found in [46,23,63,65].

In particular, the work presented in [42] develops several key technical tools to enable the hybrid primal and Meet-LWE attack. These include a new property of Babai's nearest plane algorithm regarding projection, an approximate variant of the Meet-LWE algorithm, and a locality-sensitive hashing-based near-collision finding method. The analysis incorporates both lattice and representation techniques, with detailed heuristics and experimental evidence provided. As discussed above in the hybrid dual and Meet-LWE attack, our improved Meet-LWE attack is expected to accelerate the hybrid primal Meet-LWE attack as well. Proving and analyzing the complexity of this improved version is left as future work.

Acknowledgements

This work is the result of commissioned research project supported by the affiliated institute of ETRI[2024-035]. We thank Yongha Son for the fruitful comments and discussion.

References

1. NTRU+: Compact construction of ntru using simple encoding method, available from: https://sites.google.com/view/ntruplus/home
2. SMAUG-T: the key exchange algorithm based on module-lwe and module-lwr, available from: https://kpqc.cryptolab.co.kr/smaug-t
3. Report on the security of LWE: Improved dual lattice attack (2022), https://api.semanticscholar.org/CorpusID:251600824
4. Aharonov, D., Regev, O.: Lattice problems in NP cap conp. J. ACM **52**(5), 749–765 (2005)
5. Ajtai, M., Kumar, R., Sivakumar, D.: A sieve algorithm for the shortest lattice vector problem. In: Proceedings of the Thirty-third Annual ACM Symposium on Theory of Computing. pp. 601–610 (2001)
6. Albrecht, M.R.: On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL. In: Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Proceedings, Part II. pp. 103–129 (2017)
7. Albrecht, M.R., Curtis, B.R., Deo, A., Davidson, A., Player, R., Postlethwaite, E.W., Virdia, F., Wunderer, T.: Estimate all the {LWE, NTRU} schemes! In: Catalano, D., Prisco, R.D. (eds.) Security and Cryptography for Networks - 11th International Conference, SCN 2018, Amalfi, Italy, September 5-7, 2018, Proceedings. Lecture Notes in Computer Science, vol. 11035, pp. 351–367. Springer (2018)
8. Albrecht, M.R., Göpfert, F., Virdia, F., Wunderer, T.: Revisiting the expected cost of solving usvp and applications to LWE. In: Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Proceedings, Part I. pp. 297–322 (2017)
9. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. Journal of Mathematical Cryptology **9**(3), 169–203 (2015)
10. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key Exchange—A new hope. In: 25th USENIX Security Symposium (USENIX Security 16). pp. 327–343. USENIX Association, Austin, TX (Aug 2016)
11. Aono, Y., Wang, Y., Hayashi, T., Takagi, T.: Improved progressive BKZ algorithms and their precise cost estimation by sharp simulator. In: Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Proceedings, Part I. pp. 789–819 (2016)
12. Babai, L.: On lovász' lattice reduction and the nearest lattice point problem (shortened version). In: STACS 85, 2nd Symposium of Theoretical Aspects of Computer Science, Proceedings. pp. 13–20 (1985)
13. Becker, A., Coron, J.S., Joux, A.: Improved generic algorithms for hard knapsacks. In: Paterson, K.G. (ed.) Advances in Cryptology – EUROCRYPT 2011. pp. 364–385. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
14. Becker, A., Ducas, L., Gama, N., Laarhoven, T.: New directions in nearest neighbor searching with applications to lattice sieving, pp. 10–24

15. Becker, A., Joux, A., May, A., Meurer, A.: Decoding random binary linear codes in 2n/20: How $1+1=0$ improves information set decoding. In: Pointcheval, D., Johansson, T. (eds.) Advances in Cryptology – EUROCRYPT 2012. pp. 520–536. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
16. Bernstein, D.J., Chuengsatiansup, C., Lange, T., van Vredendaal, C.: NTRU Prime: Reducing attack surface at low cost. In: Adams, C., Camenisch, J. (eds.) Selected Areas in Cryptography – SAC 2017. pp. 235–260. Springer International Publishing, Cham (2018)
17. Bi, L., Lu, X., Luo, J., Wang, K.: Hybrid dual and meet-lwe attack. In: Nguyen, K., Yang, G., Guo, F., Susilo, W. (eds.) Information Security and Privacy. pp. 168–188. Springer International Publishing, Cham (2022)
18. de Boer, K., Ducas, L., Jeffery, S., de Wolf, R.: Attacks on the AJPS mersenne-based cryptosystem. In: Lange, T., Steinwandt, R. (eds.) Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings. Lecture Notes in Computer Science, vol. 10786, pp. 101–120. Springer (2018)
19. Bos, J.W., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G., Stehlé, D.: CRYSTALS - kyber: A cca-secure module-lattice-based KEM. In: 2018 IEEE European Symposium on Security and Privacy, EuroS&P 2018, London, United Kingdom, April 24-26, 2018. pp. 353–367. IEEE (2018)
20. Boudgoust, K., Jeudy, C., Roux-Langlois, A., Wen, W.: On the hardness of Module-LWE with binary secret. In: Paterson, K.G. (ed.) Topics in Cryptology – CT-RSA 2021. pp. 503–526. Springer International Publishing, Cham (2021)
21. Boudgoust, K., Jeudy, C., Roux-Langlois, A., Wen, W.: On the hardness of module learning with errors with short distributions. Journal of Cryptology **36**(1), 1 (2023)
22. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (leveled) fully homomorphic encryption without bootstrapping. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference. p. 309–325. ITCS '12, Association for Computing Machinery, New York, NY, USA (2012)
23. Buchmann, J.A., Göpfert, F., Player, R., Wunderer, T.: On the hardness of LWE with binary error: Revisiting the hybrid lattice-reduction and meet-in-the-middle attack. In: Progress in Cryptology - AFRICACRYPT 2016 - 8th International Conference on Cryptology in Africa, Proceedings. pp. 24–43 (2016)
24. Center, K.R.: Kpqc competition round 1 and 2, available from: https://www.kpqc.or.kr/contents/03_exhibit/board.html?board_id=board_competition
25. Center, K.R.: Kpqc competition round 2, available from: https://www.kpqc.or.kr/competition_02.html
26. Chen, C., Danba, O., Stein, J., Hülsing, A., Rijneveld, J., Schanck, J.M., Schwabe, P., Whyte, W., Zhang, Z.: Algorithm specifications and supporting documentation (2019), https://api.semanticscholar.org/CorpusID:218840608
27. Chen, H., Chillotti, I., Song, Y.: Improved bootstrapping for approximate homomorphic encryption. In: Ishai, Y., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2019. pp. 34–54. Springer International Publishing, Cham (2019)
28. Chen, Y., Nguyen, P.Q.: BKZ 2.0: Better lattice security estimates. In: Lee, D.H., Wang, X. (eds.) Advances in Cryptology – ASIACRYPT 2011. pp. 1–20. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
29. Cheon, J.H., Choe, H., Hong, D., Yi, M.: SMAUG: Pushing lattice-based key encapsulation mechanisms to the limits. In: Carlet, C., Mandal, K., Rijmen, V. (eds.) Selected Areas in Cryptography – SAC 2023. pp. 127–146. Springer Nature Switzerland, Cham (2024)

30. Cheon, J.H., Han, K., Kim, A., Kim, M., Song, Y.: Bootstrapping for approximate homomorphic encryption. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2018. pp. 360–384. Springer International Publishing, Cham (2018)
31. Cheon, J.H., Hhan, M., Hong, S., Son, Y.: A hybrid of dual and meet-in-the-middle attack on sparse and ternary secret LWE. IEEE Access **7**, 89497–89506 (2019)
32. Cheon, J.H., Kim, A., Kim, M., Song, Y.: Homomorphic encryption for arithmetic of approximate numbers. In: Takagi, T., Peyrin, T. (eds.) Advances in Cryptology – ASIACRYPT 2017. pp. 409–437. Springer International Publishing, Cham (2017)
33. D'Anvers, J.P., Rossi, M., Virdia, F.: (One) Failure Is Not an Option: Bootstrapping the Search for Failures in Lattice-Based Encryption Schemes. In: Canteaut, A., Ishai, Y. (eds.) Advances in Cryptology – EUROCRYPT 2020. pp. 3–33. Springer International Publishing, Cham (2020)
34. Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal gaussians. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology – CRYPTO 2013. pp. 40–56. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
35. Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: Crystals-dilithium: A lattice-based digital signature scheme. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2018**, 238–268 (2018)
36. Ducas, L., Pulles, L.N.: Does the dual-sieve attack on learning with errors even work? In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology – CRYPTO 2023. pp. 37–69. Springer Nature Switzerland, Cham (2023)
37. Duman, J., Hövelmanns, K., Kiltz, E., Lyubashevsky, V., Seiler, G., Unruh, D.: A thorough treatment of highly-efficient NTRU instantiations. In: Boldyreva, A., Kolesnikov, V. (eds.) Public-Key Cryptography – PKC 2023. pp. 65–94. Springer Nature Switzerland, Cham (2023)
38. Espitau, T., Joux, A., Kharchenko, N.: On a dual/hybrid approach to small secret LWE. In: Bhargavan, K., Oswald, E., Prabhakaran, M. (eds.) Progress in Cryptology – INDOCRYPT 2020. pp. 440–462. Springer International Publishing, Cham (2020)
39. Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Prest, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z.: Falcon: Fast-fourier lattice-based compact signatures over ntru (2019)
40. Güneysu, T., Lyubashevsky, V., Pöppelmann, T.: Practical lattice-based cryptography: A signature scheme for embedded systems. In: Prouff, E., Schaumont, P. (eds.) Cryptographic Hardware and Embedded Systems – CHES 2012. pp. 530–547. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
41. Guo, Q., Johansson, T., Stankovski, P.: Coded-bkw: Solving lwe using lattice codes. In: Gennaro, R., Robshaw, M. (eds.) Advances in Cryptology – CRYPTO 2015. pp. 23–42. Springer Berlin Heidelberg, Berlin, Heidelberg (2015)
42. Hhan, M., Kim, J., Lee, C., Son, Y.: Let's meet ternary keys on Babai's Plane: A hybrid of lattice-reduction and meet-lwe. Cryptology ePrint Archive, Paper 2022/1473 (2022)
43. Hoffstein, J., Pipher, J., Silverman, J.H.: In: An introduction to mathematical cryptography. Springer-Verlag New York (2008)
44. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: A ring-based public key cryptosystem. In: Buhler, J.P. (ed.) Algorithmic Number Theory. pp. 267–288. Springer Berlin Heidelberg, Berlin, Heidelberg (1998)
45. van Hoof, I., Kirshanova, E., May, A.: Quantum Key Search for Ternary LWE. In: Cheon, J.H., Tillich, J.P. (eds.) Post-Quantum Cryptography. pp. 117–132. Springer International Publishing, Cham (2021)

46. Howgrave-Graham, N.: A Hybrid Lattice-Reduction and Meet-in-the-Middle Attack Against NTRU. In: Menezes, A. (ed.) Advances in Cryptology - CRYPTO 2007. pp. 150–169. Springer Berlin Heidelberg, Berlin, Heidelberg (2007)
47. Howgrave-Graham, N., Nguyen, P.Q., Pointcheval, D., Proos, J., Silverman, J.H., Singer, A., Whyte, W.: The impact of decryption failures on the security of ntru encryption. In: Boneh, D. (ed.) Advances in Cryptology - CRYPTO 2003. pp. 226–246. Springer Berlin Heidelberg, Berlin, Heidelberg (2003)
48. Kannan, R.: Minkowski's convex body theorem and integer programming. Math. Oper. Res. **12**(3), 415–440 (1987)
49. Kim, J., Park, J.H.: NTRU+: Compact construction of ntru using simple encoding method. IEEE Transactions on Information Forensics and Security **18**, 4760–4774 (2023)
50. Kirchner, P., Fouque, P.A.: An Improved BKW Algorithm for LWE with Applications to Cryptography and Lattices. In: Gennaro, R., Robshaw, M. (eds.) Advances in Cryptology – CRYPTO 2015. pp. 43–62. Springer Berlin Heidelberg, Berlin, Heidelberg (2015)
51. Langlois, A., Stehlé, D.: Worst-case to average-case reductions for module lattices. Designs, Codes and Cryptography **75**(3), 565–599 (2015)
52. Lenstra, A., Lenstra, H., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen **261**, 515–534 (1982)
53. Lindner, R., Peikert, C.: Better key sizes (and attacks) for LWE-based encryption. In: Topics in Cryptology - CT-RSA 2011 - The Cryptographers' Track at the RSA Conference 2011, Proceedings. pp. 319–339 (2011)
54. Liu, M., Nguyen, P.Q.: Solving BDD by enumeration: An update. In: Topics in Cryptology - CT-RSA 2013 - The Cryptographers' Track at the RSA Conference 2013, Proceedings. pp. 293–309 (2013)
55. Lyubashevsky, V., Nguyen, N.K., Plançon, M.: Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology – CRYPTO 2022. pp. 71–101. Springer Nature Switzerland, Cham (2022)
56. Lyubashevsky, V., Peikert, C., Regev, O.: On Ideal Lattices and Learning with Errors over Rings. In: Gilbert, H. (ed.) Advances in Cryptology – EUROCRYPT 2010. pp. 1–23. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
57. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. Journal of the ACM (JACM) **60**(6), 1–35 (2013)
58. Lyubashevsky, V., Seiler, G.: NTTRU: Truly Fast NTRU Using NTT. IACR Transactions on Cryptographic Hardware and Embedded Systems **2019**, 180–201 (May 2019)
59. May, A.: How to meet ternary LWE keys. In: Malkin, T., Peikert, C. (eds.) Advances in Cryptology – CRYPTO 2021. pp. 701–731. Springer International Publishing, Cham (2021)
60. NIST: Post-quantum cryptography, available from: https://csrc.nist.gov/Projects/post-quantum-cryptography/
61. Pouly, A., Shen, Y.: Provable dual attacks on learning with errors. In: Joye, M., Leander, G. (eds.) Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part VI. Lecture Notes in Computer Science, vol. 14656, pp. 256–285. Springer (2024)
62. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM **56**(6) (sep 2009)

63. Son, Y., Cheon, J.H.: Revisiting the hybrid attack on sparse secret LWE and application to HE parameters. In: Brenner, M., Lepoint, T., Rohloff, K. (eds.) Proceedings of the 7th ACM Workshop on Encrypted Computing & Applied Homomorphic Cryptography, WAHC@CCS 2019, London, UK, November 11-15, 2019. pp. 11–20. ACM (2019)
64. Wang, Y., Aono, Y., Takagi, T.: Hardness evaluation for search LWE problem using progressive BKZ simulator. IEICE Transactions **101-A**(12), 2162–2170 (2018)
65. Wunderer, T.: A detailed analysis of the hybrid lattice-reduction and meet-in-the-middle attack. J. Math. Cryptol. **13**(1), 1–26 (2019)