IACR Communications in Cryptology
Vol. 3, No. 1, 13 pages.

A Practical Neighborhood Search Attack on Oracle MLWE

Hongxiao Wang[1], Muhammed F. Esgin[2], Ron Steinfeld[2], Markku-Juhani O. Saarinen[3] and Siu-Ming Yiu[a,1]

[1] The University of Hong Kong, China
[2] Monash University, Australia
[3] Tampere University, Finland

Abstract. The Oracle Module Learning with Errors (Oracle MLWE) assumption, recently introduced by Liu et al. (Asiacrypt 2025), strengthens standard (Module) LWE by allowing masked linear leakages of the secret under an adversarially-chosen challenge matrix. This feature is used for the construction of new efficient primitives such as Oracle MLWE-based multi-message multi-recipient KEM/PKE (mmKEM/mmPKE) without requiring public-key well-formedness proofs. In this work, we present a practical cryptanalytic attack on Oracle MLWE, which we call a neighborhood search attack. Our attack exploits adversarially-chosen matrices (or maliciously generated public keys), together with the small ring dimension and small-norm secrets required for correctness, showing that rounding errors can be recovered via a bounded search, leading to recovery of the underlying MLWE secret. To demonstrate the effectiveness of our attack, we apply it against the Oracle MLWE-based mmKEM of Liu et al. (Asiacrypt 2025), proving that its recommended parameter sets do not achieve the claimed security level. We further implement the attack in SageMath and report concrete timings, showing that an adversary controlling a moderate number of recipients can recover other recipients' encapsulated keys within a few seconds on a standard PC under the proposed parameters, which were claimed to achieve a 128-bit security level.

Keywords: Oracle MLWE · neighborhood search attack · mmKEM

1 Introduction

Recent years have witnessed an increasing number of works (e.g. [AKSY22, WW23, KLSS23, DKL+23, ENP24, ESTX24, LSW25, BJTW26, WSS+26]) on variants of standard lattice-based cryptographic assumptions, namely (Module) Learning with Errors (LWE) and Short Integer Solutions (SIS). A natural variant of MLWE is when standard MLWE samples are accompanied by additional *hints* leaked about the MLWE secret and/or error.

The Oracle Module Learning with Errors (Oracle MLWE) assumption, recently introduced by Liu et al. [LSTW25], is such a new leakage-style strengthening of the standard (Module) LWE assumption. Roughly speaking, it considers distinguishing an MLWE instance $\mathbf{Ax} + \mathbf{e}$ even when the distinguisher additionally receives *masked* linear leakages of the secret of the form $\mathbf{A}_{\mathcal{A}}\mathbf{x} + \mathbf{e}' + \mathbf{u}$ together with hash values on the most significant part of the masking (e.g., $H(\lfloor p \cdot \mathbf{u}[i]/q \rfloor)$). A key feature that sets Oracle

E-mail: hxwang@cs.hku.hk (Hongxiao Wang), Muhammed.Esgin@monash.edu (Muhammed F. Esgin), Ron.Steinfeld@monash.edu (Ron Steinfeld), markku-juhani.saarinen@tuni.fi (Markku-Juhani O. Saarinen), smyiu@cs.hku.hk (Siu-Ming Yiu)
[a]Corresponding author.

This work is licensed under a "CC BY 4.0" license.

MLWE apart is that the challenge matrix $\mathbf{A}_{\mathcal{A}}$ can be chosen *arbitrarily* by the adversary, and consequently, Oracle MLWE is not known to admit a reduction from standard MLWE. In contrast, many other LWE-with-leakage variants, such as (Matrix) Hint-MLWE [KLSS23, ENP24, WSS+26], Leaky LWE [DKL+23, LSW25, BJTW26], and Extended LWE [ASP12, ALS16, AHPS23, WLL24], admit reductions from standard (M)LWE under suitable parameters, where the corresponding leakage matrix or linear operator (playing a role analogous to $\mathbf{A}_{\mathcal{A}}$) is required to satisfy boundedness or structural constraints.

The relaxation in Oracle MLWE enables new instantiations of advanced cryptographic schemes. In particular, Liu et al. [LSTW25] propose Oracle MLWE-based multi-message multi-recipient KEM/PKE (mmKEM/mmPKE), which batch-encrypts multiple messages for many recipients in one shot and can significantly reduce bandwidth compared to encrypting each message independently. This is especially relevant in the post-quantum setting, where ciphertexts are typically much larger than plaintexts. Moreover, compared to (Matrix) Hint-MLWE-based mmKEM/mmPKE [WSS+26], the Oracle MLWE-based approach avoids requiring recipients to provide well-formedness proofs for their public keys, thanks to the unrestricted choice of the challenge matrix $\mathbf{A}_{\mathcal{A}}$ allowed by the Oracle MLWE problem.

However, allowing adversarially-chosen challenge matrices with unbounded entries makes the resulting leakage harder to quantify and may open the door to concrete attacks via maliciously crafted matrices (or, in the mmKEM setting, maliciously generated public keys). In the concrete parameter sets of [LSTW25], correctness considerations further push the construction toward small ring dimension and small-norm secrets, which can amplify such vulnerabilities. This motivates us to conduct a careful cryptanalytic study of Oracle MLWE and its mmKEM constructions.

Our contributions. We advance the understanding of the Oracle MLWE assumption by presenting a concrete attack and its implications:

- We propose a practical cryptanalytic attack, which we call a *neighborhood search attack*, on the Oracle MLWE assumption, and provide a detailed running-time analysis.
- Using this attack, we show that the Oracle MLWE parameter regime proposed in [LSTW25] (including the parameter sets suggested for mmKEM) does not achieve the claimed security level, and we derive a revised parameter regime that may avoid our attack.
- As a direct application, we instantiate our attack to practically break the Oracle MLWE-based mmKEM of [LSTW25].
- We implement the attack and report concrete timings. As shown in Table 1, even for parameter sets designed to target a claimed 128-bit security level in [LSTW25], an adversary controlling only a moderate number of recipients can efficiently recover other recipients' encapsulated keys within *a few seconds* on a standard PC.

Table 1: Implementation results of our attack on the Oracle MLWE-based mmKEM in [LSTW25], averaged over 100 random instances on a MacBook Air (Apple M2, 3.49 GHz, 16 GB RAM), single-threaded.

Parameter Set	**Target Security Level**	**# of Controlled Recipients**	**Total Runtime**
Param1	128-bit (claimed)	48	50 s
Param2	128-bit (claimed)	128	0.2 s

At this stage, circumventing our attack (without substantially diverging from the original ideas in [LSTW25]) would require moving to a more conservative parameter

regime, e.g., by (significantly) increasing the ring dimension and/or the secret norm bound. Such changes directly incur substantially *larger parameters* (such as larger moduli) and noticeably *higher decapsulation runtime*. Consequently, pushing parameters high enough to rule out our attack would significantly undermine the original efficiency motivation behind the Oracle MLWE-based mmKEM in [LSTW25]. In fact, there already exist significantly more efficient mmPKE/mmKEM constructions [WSS+26] (concurrent work with [LSTW25]) that achieve near-optimal bandwidth performance and provable security under standard lattice assumptions, namely MLWE (via Hint-MLWE). Therefore, we do not see a clear path toward a practical fix of the mmKEM proposal in [LSTW25] at the current stage.

In order to avoid any misunderstanding, we note that our attack does not extend to the standard MLWE problem and the hint variants of MLWE that enjoy a reduction from standard MLWE (such as (Matrix) Hint-MLWE [KLSS23, ENP24]).

2 Preliminaries

2.1 Notation

Let $\lambda \in \mathbb{N}$ denote the security parameter. For a positive integer n, we denote the set $\{1, \ldots, n\}$ by $[n]$. For a positive integer q, we denote the integers modulo q by $\mathbb{Z}_q$ and define $\mathcal{R}_q := \mathbb{Z}_q[X]/(X^N + 1)$ as the quotient ring of polynomials modulo q and $X^N + 1$. We use bold lowercase letters to denote vectors of polynomial elements, e.g., $\mathbf{u} \in \mathcal{R}_q^m$, and bold uppercase letters to denote matrices of polynomial elements, e.g., $\mathbf{U} \in \mathcal{R}_q^{m \times n}$. We use $\langle \cdot, \cdot \rangle$ to denote the inner product and $\| \cdot \|_2$, $\| \cdot \|_1$, and $\| \cdot \|_\infty$ to denote the usual norms (applied coefficient-wise to polynomial elements and vectors). We denote the rounding operation by $\lfloor \cdot \rceil$, $\lfloor \cdot \rfloor$, and $\lceil \cdot \rceil$; for example, $\lfloor a \rceil$ rounds a to the nearest integer (applied coefficient-wise to polynomial elements and vectors).

We denote assignment by $:=$; for example, $x := y$ assigns the value of y to x. We denote sampling or output by $\leftarrow$; for example, $x \leftarrow \mathcal{D}$ indicates that x is sampled from the distribution $\mathcal{D}$, and $x \leftarrow \mathsf{A}(y)$ denotes that x is the output of a probabilistic polynomial-time (PPT) algorithm A on input y. In particular, we write $x \leftarrow S$ when x is sampled uniformly at random from the finite set S. We denote by $\mathsf{poly}(\lambda)$ the class of polynomial functions, i.e., $\mathsf{poly}(\lambda) = \bigcup_{c \in \mathbb{N}} O(\lambda^c)$, and by $\mathsf{negl}(\lambda)$ the class of negligible functions, i.e., $\mathsf{negl}(\lambda) = \bigcap_{c \in \mathbb{N}} o(\lambda^{-c})$.

We denote by $\mathcal{D}_\beta$ a distribution with ℓ_∞-norm bound β, i.e., $\Pr_{x \leftarrow \mathcal{D}_\beta}\big[\, \|x\|_\infty > \beta \,\big] \leqslant \mathsf{negl}(\lambda)$, for security parameter λ. We denote by $\mathcal{D}^\ell$ the ℓ-wise independent joint distribution $\mathcal{D} \times \mathcal{D} \times \cdots \times \mathcal{D}$. We denote by χ_σ a discrete Gaussian distribution with mean 0 and standard deviation σ, and let $|\chi_\sigma|$ (or $|\sigma|$) denote its ℓ_∞-norm bound, i.e., $\Pr_{x \leftarrow \chi_\sigma}\big[\, \|x\|_\infty > |\chi_\sigma| \,\big] \leqslant \mathsf{negl}(\lambda)$.

2.2 Multi-message Multi-recipient Key Encapsulation Mechanism

We recall the definition of mmKEM in [LSTW25, PPS14] as follows. Note that, as in all existing works [Kur02, BBS03, BBKS07, PPS14, KKPP20, AHK+23, LSTW25, WSS+26], mmKEM/mmPKE constructions follow a *decomposable* form, i.e., the $\mathsf{mmEncap}$ algorithm can be decomposed into a public-key-independent algorithm $\mathsf{mmEncap}^{\mathsf{i}}$ and a public-key-dependent algorithm $\mathsf{mmEncap}^{\mathsf{d}}$. We also note that mmKEM can be naturally extended to mmPKE using the generic KEM-DEM paradigm as in [PPS14].

Definition 1 (Decomposable Multi-Message Multi-Recipient KEM)**.** A decomposable mmKEM scheme with a public-private key pair space $\mathcal{K}$, an encapsulated key space $\mathcal{M}$, a multi-recipient ciphertext space $\mathcal{C}$, and an individual ciphertext space $\mathcal{C}_{\mathsf{s}}$ consists of the following algorithms:

- $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$: On input a security parameter 1^λ, it outputs a public parameter pp.
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$: On input a public parameter pp, it outputs a public-private key pair $(\mathsf{pk}, \mathsf{sk}) \in \mathcal{K}$.
- $(\mathbf{ct} := (\widehat{\mathsf{ct}}, (\widehat{\mathsf{ct}}_i)_{i\in[T]}), \mathbf{K} := (\mathsf{K}_i)_{i\in[T]}) \leftarrow \mathsf{mmEncap}(\mathsf{pp}, (\mathsf{pk}_i)_{i\in[T]}; \hat{\mathsf{r}}, (\hat{\mathsf{r}}_i)_{i\in[T]})$: On input a public parameter pp, T public keys $(\mathsf{pk}_i)_{i\in[T]}$, $(T+1)$ randomness $\hat{\mathsf{r}}$, $(\hat{\mathsf{r}}_i)_{i\in[T]}$, it can be split into two algorithms:
  - $\widehat{\mathsf{ct}} \leftarrow \mathsf{mmEncap}^{\mathsf{i}}(\mathsf{pp}; \hat{\mathsf{r}})$: On input a public parameter pp, and a randomness $\hat{\mathsf{r}}$, it outputs a public-key-*independent* ciphertext $\widehat{\mathsf{ct}}$.
  - $(\widehat{\mathsf{ct}}_i, \mathsf{K}_i) \leftarrow \mathsf{mmEncap}^{\mathsf{d}}(\mathsf{pp}, \mathsf{pk}_i; \hat{\mathsf{r}}, \hat{\mathsf{r}}_i)$: On input a public parameter pp, a public key pk_i, and randomness $\hat{\mathsf{r}}$, $\hat{\mathsf{r}}_i$, it outputs a public-key-*dependent* ciphertext $\widehat{\mathsf{ct}}_i$ and an encapsulated key K_i.
- $\mathsf{ct}_i := (\widehat{\mathsf{ct}}, \widehat{\mathsf{ct}}_i)/\perp \leftarrow \mathsf{mmExt}(i, \mathbf{ct})$: On input a multi-recipient ciphertext $\mathbf{ct} \in \mathcal{C}$, and an index $i \in [T]$, it deterministically outputs the individual ciphertext $\mathsf{ct}_i \in \mathcal{C}_\mathsf{s}$ or a symbol $\perp$ to indicate extraction failure.
- $\mathsf{K}/\perp \leftarrow \mathsf{Decap}(\mathsf{pp}, \mathsf{sk}, \mathsf{ct})$: On input a public parameter pp, a private key sk, and an individual ciphertext $\mathsf{ct} \in \mathcal{C}_\mathsf{s}$, it outputs an encapsulated key $\mathsf{K} \in \mathcal{M}$ or a symbol $\perp$ to indicate decapsulation failure.

Correctness. Let $\zeta : \mathbb{N} \to [0, 1]$. We say an mmKEM scheme is ζ-correct, if for all $\lambda, T \in \mathbb{N}$, the following probability is no more than $\zeta(\lambda)$,

$$\Pr\left[\begin{array}{c} \exists i \in [T] : \\ \mathsf{Decap}(\mathsf{pp}, \mathsf{sk}_i, \mathsf{ct}_i) \neq \mathsf{K}_i \end{array} \middle| \begin{array}{c} \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda); \\ \forall i \in [T] : \\ (\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{pp}); \\ (\mathbf{ct}, (\mathsf{K}_i)_{i\in[T]}) \leftarrow \mathsf{mmEncap}(\mathsf{pp}, (\mathsf{pk}_i)_{i\in[T]}); \\ \mathsf{ct}_i \leftarrow \mathsf{mmExt}(\mathsf{pp}, i, \mathbf{ct}) \end{array}\right].$$

Security. Let mmKEM be an mmKEM scheme, and let $\lambda \in \mathbb{N}$ be the security parameter. Let $\mathsf{ATK} \in \{\mathsf{CPA}, \mathsf{CCA}\}$. With the security game $\mathsf{GAME}^{\mathsf{mmIND\text{-}ATK}}_{\mathsf{mmKEM},T,\mathcal{A}}(\lambda)$ defined in Figure 1, we say that mmKEM is $\mathsf{mmIND\text{-}ATK}$ secure if for any PPT adversary $\mathcal{A}$, the following advantage is negligible in λ:

$$\left|\Pr[\mathsf{GAME}^{\mathsf{mmIND\text{-}ATK}}_{\mathsf{mmKEM},T,\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|.$$

3 Oracle MLWE and Its mmKEM Construction

In this section, we recall the definition of Oracle MLWE assumption and Oracle MLWE-based mmKEM constructions in [LSTW25].

Definition 2 (Decisional Oracle MLWE [LSTW25])**.** Let $n, N, q, \mathcal{D}, \chi_\sigma, p$ be parameters depending on the security parameter λ, where N is a power of two. Let $\mathcal{R} = \mathbb{Z}[X]/(X^N+1)$ and let $H : \mathcal{R}_p \to \mathcal{M}$ be a random oracle.

The oracle module learning with error (OMLWE) problem, denoted by $\mathsf{OMLWE}_{n,N,q,\mathcal{D},\chi_\sigma,p,H}$, is defined as follows. For any $w_1, w_2, w_\mathcal{A}$ bounded by a polynomial in λ, let $\mathbf{A}_1 \leftarrow \mathcal{R}_q^{w_1\times n}$ and $\mathbf{A}_2 \leftarrow \mathcal{R}_q^{w_2\times n}$ be given to a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, who

Game $\mathsf{GAME}^{\mathsf{mmIND\text{-}ATK}}_{\mathsf{mmKEM},T,\mathcal{A}}(\lambda)$ $\mathsf{ATK} \in \{\mathsf{CPA}, \mathsf{CCA}\}$

$b \leftarrow \{0,1\}$
$(\mathcal{A}_0, \mathcal{A}_1) \leftarrow \mathcal{A}$
$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
for $i \in [T]$ **do** $\mathsf{K}_i^1 \leftarrow \mathcal{M}$
for $i \in [T]$ **do** $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$
$((\mathsf{pk}_i^*)_{i\in[T]}, \mathsf{st}) \leftarrow \mathcal{A}_0(\mathsf{pp}, (\mathsf{pk}_i)_{i\in[T]})$
$(\mathbf{ct}, \mathbf{K}^0) \leftarrow \mathsf{mmEncap}(\mathsf{pp}, (\mathsf{pk}_i^*)_{i\in[T]})$
for all $i \in [T]$
  if $\mathsf{pk}_i^* \notin (\mathsf{pk}_i)_{i\in[T]}$ **do** $\mathsf{K}_i^1 := \mathsf{K}_i^0$
end for
$b' \leftarrow \mathcal{A}_1(\mathbf{ct}, \mathbf{K}^b, \mathsf{st})$
return $[b = b']$

Oracle $\mathbf{Dec}_0(i, \mathsf{ct})$

req: $i \in [T]$
return $\mathsf{K} \leftarrow \mathsf{Decap}(\mathsf{pp}, \mathsf{sk}_i, \mathsf{ct})$

Oracle $\mathbf{Dec}_1(i, \mathsf{ct})$

req: $i \in [T]$
req: $\mathsf{ct} \neq \mathsf{mmExt}(\mathbf{ct}, i)$
return $\mathsf{K} \leftarrow \mathsf{Decap}(\mathsf{pp}, \mathsf{sk}_i, \mathsf{ct})$

Figure 1: The mmIND-ATK security games for mmKEM with $\mathsf{ATK} \in \{\mathsf{CPA}, \mathsf{CCA}\}$. If $\mathsf{ATK} = \mathsf{CCA}$, the adversary $\mathcal{A}_0$, $\mathcal{A}_1$ can have the access to the decryption oracle $\mathbf{Dec}_0$ and $\mathbf{Dec}_1$ respectively.

then chooses $\mathbf{A}_\mathcal{A} \in \mathcal{R}_q^{w_\mathcal{A} \times n}$ *arbitrarily*. The challenger then asks $\mathcal{A}$ to distinguish between the following two cases (difference marked in blue):

$$\Big(\mathbf{A}_2,\ \mathbf{A}_2\mathbf{x} + \mathbf{e}_2 + \mathbf{u}_2,\ \big(H(\lfloor p \cdot u_2^{(i)}/q \rceil)\big)_{i\in[w_2]}\Big),$$
$$\Big(\mathbf{A}_1,\ \mathbf{A}_1\mathbf{x} + \mathbf{e}_1\Big),$$
$$\Big(\mathbf{A}_\mathcal{A},\ \mathbf{A}_\mathcal{A}\mathbf{x} + \mathbf{e}_\mathcal{A} + \mathbf{u}_\mathcal{A},\ \big(H(\lfloor p \cdot u_\mathcal{A}^{(i)}/q \rceil)\big)_{i\in[w_\mathcal{A}]}\Big)$$

from

$$\Big(\mathbf{A}_2,\ \mathbf{r}_2,\ \big(H(\lfloor p \cdot u_2^{(i)}/q \rceil)\big)_{i\in[w_2]}\Big),$$
$$\Big(\mathbf{A}_1,\ \mathbf{A}_1\mathbf{x} + \mathbf{e}_1\Big),$$
$$\Big(\mathbf{A}_\mathcal{A},\ \mathbf{A}_\mathcal{A}\mathbf{x} + \mathbf{e}_\mathcal{A} + \mathbf{u}_\mathcal{A},\ \big(H(\lfloor p \cdot u_\mathcal{A}^{(i)}/q \rceil)\big)_{i\in[w_\mathcal{A}]}\Big).$$

Here, the following values are sampled by the challenger:

$$\mathbf{x} \leftarrow \mathcal{D}^n, \quad \mathbf{e}_1 \leftarrow \chi_\sigma^{w_1}, \quad \mathbf{e}_2 \leftarrow \chi_\sigma^{w_2}, \quad \mathbf{e}_\mathcal{A} \leftarrow \chi_\sigma^{w_\mathcal{A}}, \quad \mathbf{r}_2 \leftarrow \mathcal{R}^{w_2},$$
$$\mathbf{u}_2 = (u_2^{(1)}, \ldots, u_2^{(w_2)}) \leftarrow \mathcal{R}_q^{w_2}, \quad \mathbf{u}_\mathcal{A} = (u_\mathcal{A}^{(1)}, \ldots, u_\mathcal{A}^{(w_\mathcal{A})}) \leftarrow \mathcal{R}_q^{w_\mathcal{A}}.$$

We say that $\mathsf{OMLWE}_{n,N,q,\mathcal{D},\chi_\sigma,p,H}$ is hard if, for any PPT adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ in distinguishing the above two cases is negligible in λ.[1]

Compared to the (Matrix) Hint-MLWE assumption [KLSS23, ENP24, WSS+26], which can be reduced from the standard MLWE assumption under suitable parameters, the main difference between Oracle MLWE and (Matrix) Hint-MLWE is that Oracle MLWE does *not* require the challenge matrix $\mathbf{A}_\mathcal{A}$ to be norm-bounded (small), e.g., $\|\mathbf{A}_\mathcal{A}\| \leqslant \beta$.

Looking ahead to Oracle MLWE-based mmKEM constructions, the advantage of this relaxation in Oracle MLWE assumption is that public keys in its mmPKE/KEM scheme need not be proven well-formed, whereas in Matrix Hint-MLWE-based mmPKE/KEM schemes [WSS+26], such proofs are required, albeit only once. However, a major drawback of the Oracle MLWE assumption is that it does not admit a reduction from standard

[1] The definition can be extended to more general noise distributions. For simplicity, and following [LSTW25], we focus on Gaussian noise.

MLWE and, as we will demonstrate later, is susceptible to concrete attacks. Furthermore, since there is no restriction on the adversary-chosen matrix $\mathbf{A}_{\mathcal{A}}$ (unlike in the case of the (Matrix) Hint-MLWE assumption), it becomes difficult to quantify how much information about the secret $\mathbf{x}$ is leaked by the hints $(\mathbf{A}_{\mathcal{A}}\mathbf{x} + \mathbf{e}_{\mathcal{A}} + \mathbf{u}_{\mathcal{A}}, (H(\lfloor p \cdot u_{\mathcal{A}}^{(i)}/q \rceil))_{i\in[w_{\mathcal{A}}]})$.

3.1 Oracle MLWE-Based mmKEM

An mmKEM construction can be naturally instantiated from Oracle MLWE. The work [LSTW25] presents two constructions of mmKEM. The first construction employs a super-polynomial modulus to guarantee correctness. The second construction reduces the modulus from super-polynomial to polynomial size, at the cost that the receiver may need to repeatedly decrypt until the correct result is obtained. We recall these constructions as follows. For the formal correctness and security analyses, we refer the reader to [LSTW25].

Construction 3 (Super-Polynomial-Modulus Oracle MLWE-Based mmKEM [LSTW25])**.** Let λ be a security parameter. Let $\mathcal{M}$ be an encapsulated key space such that $|\mathcal{M}| \geqslant 2^{\lambda}$. Let ℓ, p be positive integers, such that $\ell := \lceil \max(\log(|\mathcal{M}|, \lambda)/\log p)$. Let the ring dimension $N \geqslant \ell$, private key length n, private key distribution $\mathcal{D}_{\beta}$, Gaussian noise with standard deviation σ, and uniform noise bound r, such that $(N \cdot n \cdot |\chi_{\sigma}| \cdot |\mathcal{D}_{\beta}|)/r = \mathsf{negl}(\lambda)$. Let the ciphertext modulus $q := p \cdot r$. Let $H : \mathcal{R}_p \to \mathcal{M}$ be a hash function. The detailed construction is shown as follows.

- $\mathsf{pp} \leftarrow \mathsf{Setup}(1^{\lambda})$: On input a security parameter λ, it samples the shared common polynomial matrix $\mathbf{A} \leftarrow \mathcal{R}_q^{n\times n}$ and outputs the public parameters $\mathsf{pp} := (\mathbf{A}, H)$.
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$: On input the public parameters $\mathsf{pp} = (\mathbf{A}, H)$, it first samples $\mathbf{s} \in \mathcal{R}^n \leftarrow \mathcal{D}_{\beta}^n$, $\mathbf{e} \in \mathcal{R}^n \leftarrow \chi_{\sigma}^n$ and then computes the public key as,
$$\mathbf{b} := \mathbf{A}^{\top}\mathbf{s} + \mathbf{e}. \tag{1}$$
It outputs the public-private key pair $(\mathsf{pk} := \mathbf{b},\ \mathsf{sk} := \mathbf{s})$.
- $(\mathbf{ct} := (\widehat{\mathsf{ct}}, (\widehat{\mathsf{ct}}_i)_{i\in[T]}), \mathbf{K} := (\mathsf{K}_i)_{i\in[T]}) \leftarrow \mathsf{mmEncap}(\mathsf{pp}, (\mathsf{pk}_i)_{i\in[T]})$: On input the public parameter $\mathsf{pp} = (\mathbf{A}, H)$, a set of public keys $(\mathsf{pk}_i)_{i\in[T]} = (\mathbf{b}_i)_{i\in[T]}$, it first samples $\mathbf{x} \in \mathcal{R}^n \leftarrow \mathcal{D}_{\beta}^n$, $\mathbf{e}_u \in \mathcal{R}^n \leftarrow \chi_{\sigma}^n$ and then computes the public-key-independent ciphertext as,
$$\mathbf{c} := \mathbf{A}\mathbf{x} + \mathbf{e}_u. \tag{2}$$
Then, for all $i \in [T]$, it first samples $u_i \leftarrow \mathcal{R}_p$, $y_i \leftarrow \mathcal{R}_r$, and $e_i' \in \mathcal{R} \leftarrow \chi_{\sigma}$, and then computes the public-key-dependent ciphertext as,
$$v_i := \langle \mathbf{b}, \mathbf{x} \rangle + r \cdot u_i + y_i + e_i'. \tag{3}$$
Next, for all $i \in [T]$, it computes the encapsulated key as $k_i \leftarrow H(u_i)$.
Last, it sets $\widehat{\mathsf{ct}} := \mathbf{c}$, and $\widehat{\mathsf{ct}}_i := v_i$, $\mathsf{K}_i := k_i$ for all $i \in [T]$ and outputs the multi-recipient ciphertext $\mathbf{ct} := (\widehat{\mathsf{ct}}, (\widehat{\mathsf{ct}}_i)_{i\in[T]})$ and the encapsulated key $\mathbf{K} := (\mathsf{K}_i)_{i\in[T]}$.
- $\mathsf{ct}_i \leftarrow \mathsf{mmExt}(\mathbf{ct}, i)$: On input the multi-recipient ciphertext $\mathbf{ct} = (\mathbf{c}, (v_i)_{i\in[T]})$ and an index $i \in [T]$, it outputs the individual ciphertext $\mathsf{ct}_i := (\mathbf{c}, v_i)$.
- $\mathsf{K} \leftarrow \mathsf{Decap}(\mathsf{pp}, \mathsf{ct}, \mathsf{sk})$: On input the public parameter $\mathsf{pp} = (\mathbf{A}, H)$, an individual ciphertext $\mathsf{ct} = (\mathbf{c}, v)$ and a private key $\mathsf{sk} = \mathbf{s}$, it computes the encapsulated key as,
$$u := \left\lfloor \frac{v_i - \langle \mathbf{c}, \mathbf{s} \rangle}{r} \right\rceil, \qquad k_i := H(u). \tag{4}$$
It outputs the encapsulated key $\mathsf{K} := k$.

The security of Construction 3 is based on the Oracle MLWE assumption. In particular, in mmEncap algorithm, the public keys $(\mathbf{b}_i)_{i\in[T]}$ compose the challenge matrix $\mathbf{A}_{\mathcal{A}}$ in Definition 2, the term $r \cdot u_i + y_i$ in Equation (3), where $q = p \cdot r$ and $u_i \leftarrow \mathcal{R}_p$ and $y_i \leftarrow \mathcal{R}_r$ are sampled uniformly at random, serving as the masking term $u_{\mathcal{A}}^{(i)} \in \mathcal{R}_q$ as defined in Definition 2. Each $k_i = H(u_i)$ corresponds to $H(\lfloor p \cdot u_{\mathcal{A}}^{(i)}/q \rfloor)$ in Definition 2.

To decapsulate an individual ciphertext $(\mathbf{c}, v_i)$, the recipient uses the private key $\mathbf{s}_i$ to compute $v_i - \langle \mathbf{c}, \mathbf{s}_i \rangle$, as specified by the Decap algorithm. Substituting Equations (1), (2), and (3) yields

$$v_i - \langle \mathbf{c}, \mathbf{s}_i \rangle = \langle -\mathbf{s}_i \| \mathbf{e}_i, \mathbf{e}_u \| \mathbf{x} \rangle + e'_i + y_i + r \cdot u_i, \tag{5}$$

where $\|$ denotes concatenation. Here, we denote the decryption error by

$$h_i := \langle -\mathbf{s}_i \| \mathbf{e}_i, \mathbf{e}_u \| \mathbf{x} \rangle + e'_i + y_i.$$

If $\|h_i\|_\infty \leqslant r$, then decryption is correct and u_i can be recovered as $u_i := \lfloor (v_i - \langle \mathbf{c}, \mathbf{s}_i \rangle)/r \rceil$, together with the encapsulated key $H(u_i)$.

However, a fundamental issue is that y_i is sampled uniformly over $\mathcal{R}_r$, so $\|y_i\|_\infty$ may be as large as r, making $\|h_i\|_\infty \geqslant r$ and causing decryption to fail. To address this issue, the work [LSTW25] first considers using a super-polynomial modulus q, as in Construction 3, such that $(N \cdot n \cdot |\chi_\sigma| \cdot |\mathcal{D}_\beta|)/r = \mathsf{negl}(\lambda)$, that is, $\|\langle -\mathbf{s}_i \| \mathbf{e}_i, \mathbf{e}_u \| \mathbf{x} \rangle + e'_i\|_\infty / r \leqslant \mathsf{negl}(\lambda)$. This implies that $\Pr[\|h_i\|_\infty \geqslant r] \leqslant \mathsf{negl}(\lambda)$. However, such a super-polynomial modulus is clearly impractical.

Subsequently, the work [LSTW25] modifies Construction 3 by using a polynomial modulus q such that $(N \cdot n \cdot |\chi_\sigma| \cdot |\mathcal{D}_\beta|)/r \leqslant 1/\lambda$. In this case, the decryption error is bounded by $\|h_i\|_\infty < 2r$, so the decryption result may differ by *at most one* (i.e., a coefficient-wise shift by one) when $\|h_i\|_\infty \geqslant r$. Consequently, the multi-recipient ciphertext is augmented with additional hash values $c_i := G(u_i)$ using an additional hash function $G : \mathcal{R}_p \to \{0,1\}^\lambda$; that is, the multi-recipient ciphertext becomes $(\mathbf{c}, (v_i, c_i)_{i\in[T]})$. This enables the recipient to iteratively test candidate values u'_i by shifting coefficients by one until $c_i = G(u'_i)$ holds. The work [LSTW25] estimates the number of such tests to be $O(\mathsf{poly}(\log \lambda))$. However, this number may *increase substantially*—up to $O(2^N)$—if optimizations such as *modulus switching* are applied, since such techniques introduce additional decryption error. For further details of the polynomial-modulus construction of the Oracle MLWE-based mmKEM, we refer the reader to [LSTW25].

Finally, we list the parameter sets provided in [LSTW25] in Table 2. We note that both Param1 and Param2 are designed to target a 128-bit security level. The main difference is that Param1 (with a smaller modulus q) favors a smaller ciphertext size, whereas Param2 (with a smaller ring dimension N) is intended to achieve faster decryption.

Table 2: Parameter sets for the Oracle MLWE-based mmKEM [LSTW25] targeting 128-bit security, together with the estimated complexity of our attack.

Notation	**Description**	Param1	Param2
n	Dimension of the public matrix $\mathbf{A}$	48	128
q	Modulus of the public matrix and ciphertext	2^{25}	2^{41}
N	Polynomial degree of the ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^N + 1)$	16	8
σ	Standard deviation of the Gaussian noise for $\mathbf{e}_u$ and e'_i	2^8	2^{16}
β	ℓ_∞-norm bound of secret vectors $\mathbf{s}$ and $\mathbf{x}$	1	1
h	Hamming weight of binary secret vectors $\mathbf{s}$ and $\mathbf{x}$	280	350
p	Modulus used for u_i	2^8	2^{16}
r	Modulus used for y_i	2^{17}	2^{25}
	Claimed security level in [LSTW25]	128-bit	128-bit
	Estimated attack complexity	$\approx 2^{31}$	$\approx 2^{20}$

4 Our Neighborhood Search Attack

In this section, we present a cryptanalytic attack, which we term a *neighborhood search attack*, against the Oracle MLWE assumption. Based on this attack, we derive a revised parameter regime for Oracle MLWE. Moreover, as a direct application, we show how the Oracle MLWE-based mmKEM of [LSTW25] can be broken, and report concrete implementation results.

We begin by providing intuition for our attack.

Intuition. We first observe that Oracle MLWE and its mmKEM in [LSTW25] exhibit several properties that are relevant to our attack:

- Oracle MLWE (or its mmKEM) allows an adversary (or recipients) to generate the challenge matrix $\mathbf{A}_{\mathcal{A}}$ (or their own public keys $\mathbf{b}_i$) *arbitrarily*, without any restriction such as a well-formedness proof.
- The polynomial ring dimension N used in the parameter sets is relatively small (e.g., 8 or 16), since Oracle MLWE-based mmKEM with a polynomial modulus must ensure efficient decryption even in the *worst case*, i.e., $N \ll \lambda$, as discussed earlier.
- The ℓ_∞-norm bound β of the secret vector $\mathbf{x}$ in the ciphertext is small (e.g., $\mathbf{x} \in \{0,1\}^{dn}$), so as to minimize the decryption error and thereby reduce the modulus q.

With these observations, an adversary (or a recipient) can maliciously generate public keys (e.g., by setting their private keys to $(\gamma, 0, \ldots, 0)$ for a carefully chosen value γ). This enables the attacker to obtain a masked value that approximates the true encapsulated value u_i up to the decryption error h_i. Importantly, this error is determined only by the noise distribution and is independent of the secret dimension n. Thus, the attacker can recover the correct value of u_i by searching over a small neighborhood determined by the possible values of h_i, whose size is exponential only in the ring degree N, rather than in n.

Consequently, the attacker can not only *efficiently* obtain the encapsulated key u_i, but also recover the corresponding decryption error h_i, thereby recovering the secret $\mathbf{x}$ used in the multi-recipient ciphertext.

Our neighborhood search attack on Oracle MLWE. We show how an adversary $\mathcal{A}$ can break the Oracle MLWE assumption $\mathsf{OMLWE}_{n,N,q,\mathcal{D},\chi_\sigma,p,H}$ using a neighborhood search attack. An explicit algorithmic description of the instantiation of this attack to break the Oracle MLWE-based mmKEM is provided in Section 4.1 (see Algorithm 1), along with concrete implementation results.

We first assume that $p \ll q$. After the challenger samples $\mathbf{A}_1 \leftarrow \mathcal{R}_q^{w_1 \times n}$ and $\mathbf{A}_2 \leftarrow \mathcal{R}_q^{w_2 \times n}$, the adversary $\mathcal{A}$ sets $w_{\mathcal{A}} := n$ and constructs the challenge matrix $\mathbf{A}_{\mathcal{A}}$ as

$$\mathbf{A}_{\mathcal{A}} := \begin{bmatrix} \mathbf{b}_1^\top \\ \vdots \\ \mathbf{b}_n^\top \end{bmatrix} \in \mathcal{R}_q^{n \times n}, \quad \text{where} \quad \mathbf{b}_i := \mathbf{A}_1^\top \hat{\mathbf{s}}_i + \hat{\mathbf{e}}_i \in \mathcal{R}_q^n \quad \text{for all } i \in [n].$$

Here, the adversary $\mathcal{A}$ sets $\hat{\mathbf{s}}_i = (\hat{s}_i^{(1)}, \ldots, \hat{s}_i^{(w_1)}) \in \mathcal{R}_q^{w_1}$ with $\hat{s}_i^{(1)} := 1 \in \mathbb{Z}_q$ and $\hat{s}_i^{(j)} := 0$ for all $j \in [w_1] \backslash \{1\}$, and sets $\hat{\mathbf{e}}_i = (\hat{e}_i^{(1)}, \ldots, \hat{e}_i^{(n)}) \in \mathcal{R}_q^n$ with $\hat{e}_i^{(i)} := \gamma \in \mathbb{Z}_q$ and $\hat{e}_i^{(j)} := 0$ for all $j \in [n] \backslash \{i\}$, where $\gamma \geqslant \lceil q/p \rceil + 2 \cdot |\chi_\sigma|$. Note that $\hat{\mathbf{s}}_i$ can be chosen as other short vectors, with a corresponding adjustment of γ. We use binary vectors of Hamming weight one solely for simplicity.

Let $\mathbf{x} = (x_1, \ldots, x_n) \in \mathcal{R}_q^n$, $\mathbf{e}_{\mathcal{A}} = (e_{\mathcal{A}}^{(1)}, \ldots, e_{\mathcal{A}}^{(n)}) \in \mathcal{R}_q^n$, $\mathbf{e}_1 = (e_1^{(1)}, \ldots, e_1^{(w_1)}) \in \mathcal{R}_q^{w_1}$, and $\mathbf{u}_{\mathcal{A}} = (u_{\mathcal{A}}^{(1)}, \ldots, u_{\mathcal{A}}^{n}) \in \mathcal{R}_q^n$. Define $u_i := \lfloor p \cdot u_{\mathcal{A}}^{(i)}/q \rceil$ and $y_i := u_{\mathcal{A}}^{(i)} - \lfloor q \cdot u_i/p \rceil$. Here, u_i

denotes the actual input to the hash function H, which is the value targeted by our search, while y_i denotes the residual term introduced by the rounding. This decomposition allows us to separate the hash-relevant part from the rounding error in the subsequent analysis. Additionally, we observe that $\|y_i\|_\infty \leqslant \lceil q/p \rceil$.

Next, define $\mathbf{c} := \mathbf{A}_1\mathbf{x} + \mathbf{e}_1 \in \mathcal{R}_q^{w_1}$ and

$$\begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix} := \mathbf{A}_{\mathcal{A}}\mathbf{x} + \mathbf{e}_{\mathcal{A}} + \mathbf{u}_{\mathcal{A}} \in \mathcal{R}_q^n, \quad (k_i)_{i\in[n]} := \big(H(\lfloor p \cdot u_{\mathcal{A}}^{(i)}/q \rceil)\big)_{i\in[n]}.$$

After receiving $(\mathbf{c}, (v_i, k_i)_{i\in[n]})$, for each $i \in [n]$, the adversary computes $v_i - \langle \mathbf{c}, \hat{\mathbf{s}}_i \rangle$. As in Equation (5), we have

$$v_i - \langle \mathbf{c}, \hat{\mathbf{s}}_i \rangle = \gamma \cdot x_i - e_1^{(1)} + e_{\mathcal{A}}^{(i)} + \lfloor q \cdot u_i/p \rceil + y_i. \tag{6}$$

The adversary then performs a *neighborhood search* by computing a tentative value $u_i' := \lfloor p \cdot (v_i - \langle \mathbf{c}, \hat{\mathbf{s}}_i \rangle)/q \rceil$ and checking whether $H(u_i') = k_i$. If the check fails, the adversary iteratively *shifts* the coefficients of u_i' by values in $\{0, 1, \ldots, \tau\}$ until a match is found, where τ denotes the search upper bound.

Since $p \ll q$ and $\gamma \geqslant \lceil q/p \rceil + 2 \cdot |\chi_\sigma|$, we have $\lceil q/p \rceil \gg |\mathcal{D}|, |\chi_\sigma|$, and the rounding error satisfies

$$\left\|\gamma \cdot x_i - e_1^{(1)} + e_{\mathcal{A}}^{(i)} + y_i\right\|_\infty < |\mathcal{D}| \cdot \gamma + \lceil q/p \rceil + 2 \cdot |\chi_\sigma| < (|\mathcal{D}| + 1) \cdot \gamma < (|\mathcal{D}| + 2) \cdot \lceil q/p \rceil.$$

Hence, each coefficient may be shifted by at most $|\mathcal{D}| + 1$, and the search bound satisfies $\tau \leqslant |\mathcal{D}| + 1$. As a result, the adversary needs to test at most $(|\mathcal{D}| + 2)^N$ candidates.

After recovering the correct u_i for all $i \in [n]$, since $\gamma \geqslant \| - e_1^{(1)} + e_{\mathcal{A}}^{(i)} + y_i\|_\infty$ in Equation (6), the adversary can recover each secret x_i by computing $\lfloor (v_i - \langle \mathbf{c}, \hat{\mathbf{s}}_i \rangle - \lfloor q \cdot u_i/p \rceil)/\gamma \rceil$.

Finally, let $\mathbf{u}_2 = (u_2^{(1)}, \ldots, u_2^{(w_2)}) \in \mathcal{R}_q^{w_2}$. Define

$$\begin{bmatrix} v_1' \\ \vdots \\ v_{w_2}' \end{bmatrix} := \mathbf{A}_2\mathbf{x} + \mathbf{e}_2 + \mathbf{u}_2 \in \mathcal{R}_q^{w_2}, \quad (k_i')_{i\in[w_2]} := \big(H(\lfloor p \cdot u_2^{(i)}/q \rceil)\big)_{i\in[w_2]}.$$

For each $i \in [w_2]$, the adversary computes $\hat{u}_i := v_i' - \mathbf{A}_2\mathbf{x}$ using the recovered $\mathbf{x} = (x_1, \ldots, x_n)$ and checks whether $k_i' = H(\lfloor p \cdot \hat{u}_i/q \rceil)$, which allows $\mathcal{A}$ to distinguish the Oracle MLWE instance from uniform values.

Overall, the running time of the adversary $\mathcal{A}$ is $O\big(n \cdot (|\mathcal{D}| + 2)^N\big)$. We summarize the above attack in Corollary 4.

Corollary 4. *When $p \ll q$, there exists an adversary $\mathcal{A}$ that breaks $\mathsf{OMLWE}_{n,N,q,\mathcal{D},\chi_\sigma,p,H}$ with running time $O\big(n \cdot (|\mathcal{D}| + 2)^N\big)$.*

Our attack shows that the original Regime 4 in [LSTW25, Table 2] is insecure. Accordingly, we present revised Oracle MLWE parameter regimes, i.e., Regime 5, in Table 3. As in [LSTW25], we treat $\mathcal{D}$ as a distribution bounded by β and χ_σ as a Gaussian distribution with standard deviation σ, for simplicity. We note that Regime 4 is the one used for the Oracle MLWE-based mmKEM constructions and is also the primary focus of our revision.

4.1 Breaking Oracle MLWE-Based mmKEM

Algorithm 1 presents a concrete instantiation of our neighborhood search attack to break the (polynomial-modulus) Oracle MLWE-based mmKEM proposed in [LSTW25].

Algorithm 1 Attack on Oracle MLWE-Based mmKEM [LSTW25]

procedure $\mathsf{MalKGen}(\mathsf{pp} = (\mathbf{A}, H, G), i)$
  Set $\hat{\mathbf{s}}_i := (1, 0, \ldots, 0) \in \mathcal{R}_q^n$
  Set $\hat{\mathbf{e}}_i := (\hat{e}_i^{(0)}, \ldots, \hat{e}_i^{(n)}) \in \mathcal{R}_q^n$, where $\hat{e}_i^{(i)} := r + 2 \cdot |\sigma|$ and $\hat{e}_i^{(j)} := 0$ for $j \in [n]/\{i\}$
  Compute $\hat{\mathbf{b}}_i := \mathbf{A}^\top \hat{\mathbf{s}}_i + \hat{\mathbf{e}}_i$
  return $(\mathsf{pk}_i := \hat{\mathbf{b}}_i, \mathsf{sk}_i := \hat{\mathbf{s}}_i)$
end procedure

procedure $\mathsf{PartRec}(\mathsf{pp} = (\mathbf{A}, H, G), \mathsf{ct}_i = (\mathbf{c}, v_i, c_i), \mathsf{sk}_i = \hat{\mathbf{s}}_i)$
  Compute $u_i' := \lfloor (v_i - \langle \mathbf{c}, \hat{\mathbf{s}}_i \rangle)/r \rceil$
  Iteratively shift the coefficients of u_i' by $\{0, 1, \ldots, \beta + 1\}$ until $G(u_i') = c_i$
  Compute $x_i := \lfloor (v_i - \langle \mathbf{c}, \hat{\mathbf{s}}_i \rangle - r \cdot u_i')/(r + 2 \cdot |\sigma|) \rceil$
  return x_i
end procedure

procedure $\mathsf{RecEnKey}(\mathsf{pp} = (\mathbf{A}, H, G), \mathbf{x} = (x_1, \ldots x_n), \mathsf{ct}_i = (\mathbf{c}, v_i, c_i))$
  Compute $u_i = \lfloor (v_i - \mathbf{A}\mathbf{x})/r \rceil$
  if $c_i = G(u_i)$ **then**
    Set $k_i := H(u_i)$
  else
    Set $k_i := \perp$
  end if
  return $\mathsf{K}_i := k_i$
end procedure

Table 3: Revised Oracle MLWE parameter regimes from [LSTW25]. Updated parameters are highlighted in blue.

Param. Regime	n	N	q	β for $\mathcal{D}$	σ for χ_σ	p	Attack Complexity†
1	$\mathsf{poly}(\lambda)$	$O(1)$	Any	$\mathsf{poly}(\lambda)$ Any	$\mathsf{poly}(\lambda)$, $\ll q$	$\ll q$	$(\lvert\mathcal{D}\rvert \cdot \lvert\chi_\sigma\rvert)^N$ $(\lvert\chi_\sigma\rvert)^{2N}$
2	$\mathsf{poly}(\lambda)$	$\mathsf{poly}(\lambda)$	$\mathsf{poly}(\lambda)$ and $\mathcal{R}_q$ fully splits	Any	$\ll q$	$\leqslant q$	q
3	$\mathsf{poly}(\lambda)$	$1/\lvert\sigma\rvert^N = \mathsf{neg}(\lambda)$	Any	Any	$1/\lvert\sigma\rvert^N = \mathsf{neg}(\lambda)$	$= q$	= MLWE
4	$\mathsf{poly}(\lambda)$	$1/\lvert\sigma\rvert^N = \mathsf{neg}(\lambda)$	$\mathcal{R}_q$ does not split at all	Any	$1/\lvert\sigma\rvert^N = \mathsf{neg}(\lambda)$	$\ll q$	$(2 + \lvert\mathcal{D}\rvert)^N$
5	$\mathsf{poly}(\lambda)$	$1/\lvert\sigma\rvert^N = \mathsf{neg}(\lambda)$, $1/(2 + \lvert\beta\rvert)^N = \mathsf{neg}(\lambda)$	$\mathcal{R}_q$ does not split at all	$1/(2+\beta)^N = \mathsf{neg}(\lambda)$	$1/\lvert\sigma\rvert^N = \mathsf{neg}(\lambda)$	$\ll q$	Unclear‡

† Attack complexity refers to the running time of the best-known attack for each parameter regime.
‡ Here, "Unclear" indicates that for Regime 5 all known attacks run in super-polynomial time in λ, but no reduction from the standard MLWE assumption is known.

Specifically, the attacker controls n recipients, where n matches the dimension of the public matrix as given in Table 2. This choice is made deliberately to enable the attacker to break other recipients' ciphertexts and recover their encapsulated keys.

First, after receiving the public parameters $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, each controlled recipient $i \in [n]$ executes the sub-algorithm $\mathsf{MalKGen}$ to maliciously generate its own public-private key pair. They send the public keys to the encryptor and the corresponding private keys to the attacker. We note that it is difficult for the encryptor to distinguish such malicious public keys.

Then, after receiving the multi-recipient ciphertext $(\mathbf{c}, (v_i)_{i\in[T]}, (c_i)_{i\in[T]}) \leftarrow \mathsf{mmEncap}\big(\mathsf{pp}, (\mathsf{pk}_i)_{i\in[T]}\big)$, where $(\mathsf{pk}_i)_{i\in[T]}$ contains both the maliciously generated public keys and

the honest recipients' public keys, the attacker runs the sub-algorithm PartRec to recover each element x_i of the randomness $\mathbf{x} = (x_1, \ldots, x_n)$ used in the multi-recipient ciphertext.

Finally, the attacker breaks other honest recipients' ciphertexts by running the sub-algorithm RecEnKey and recovers their encapsulated keys.

We note that our attack can also be applied to the super-polynomial-modulus Oracle MLWE-based mmKEM/PKE. Specifically, in the sub-algorithm PartRec, although the attacker cannot use c_i and $G(\cdot)$ to test the correctness of u_i, in practice, the attacker can use a tentative encapsulated key k_i' derived from u_i' to decrypt the subsequently encrypted symmetric-key data (e.g., via Authenticated Encryption with Associated Data (AEAD)) to verify the correct u_i. The asymptotic attack complexity remains $O(n \cdot (\beta + 2)^N)$, with some additional constant-factor overhead.

To test the correctness and practicality of our attack, we implemented it in SageMath[2] The results are shown in Table 1. We emphasize that both Param1 and Param2 are designed in [LSTW25] to target a 128-bit security level. For these parameter sets, if an adversary controls 48 recipients in Param1 (or 128 recipients in Param2), it can recover other recipients' encapsulated keys within about 50 s (Param1) and 0.2 s (Param2) on a MacBook Air (Apple M2, 3.49 GHz, 16 GB RAM), single-threaded. The reported timings are averaged over 100 random instances.

5 Conclusion and Discussion

In this work, we presented a practical cryptanalytic attack on the Oracle MLWE assumption, which we term a *neighborhood search attack*. The attack exploits the combination of adversarially-controlled matrices (or maliciously generated public keys) and the use of small ring dimension and small-norm secrets required for correctness. As a result, the rounding error in Oracle MLWE can be efficiently resolved via a bounded search, leading to a practical recovery of the underlying secret.

As a direct application, we instantiated our attack against the Oracle MLWE-based mmKEM of [LSTW25]. Our results show that the original parameter sets proposed therein fall far below the intended security level. We further validated our analysis with concrete implementations, demonstrating that an adversary controlling a moderate number of recipients can efficiently recover other recipients' encapsulated keys in the order of seconds under their recommended parameters.

A natural countermeasure is to move to a more conservative parameter regime, for example by significantly increasing the ring dimension N and/or the secret norm bound β, so that the neighborhood search becomes infeasible. However, such changes directly lead to much larger moduli and noticeably higher decapsulation cost, thereby undermining the original efficiency motivation of Oracle MLWE-based mmKEMs. Consequently, our results suggest that, at the current stage, Oracle MLWE is better viewed as a theoretical abstraction, while identifying practically efficient and secure constructions based on this assumption—or a practical fix of the mmKEM proposal in [LSTW25]—remains an interesting open problem.

Acknowledgements

Siu-Ming Yiu was supported by Shenzhen-Hong Kong-Macao Science and Technology Plan Project (Category C Project: SGDX20210823103537030), and the Theme-based Research Scheme of the Research Grants Council of Hong Kong (T35-710/20-R). This work was also partially supported by an Australian Research Council Discovery Grants DP250100229 and DP260100245. We thank Yunhao Wang and Zeyu Liu for helpful discussions on Oracle

[2]Implementation: https://github.com/dingdangdangwhx/OracleMLWE-mmKEM-Cryptanalysis.

MLWE and its mmKEM construction. We thank the anonymous reviewers for their helpful comments.

References

[AHK+23] Joël Alwen, Dominik Hartmann, Eike Kiltz, Marta Mularczyk, and Peter Schwabe. Post-quantum multi-recipient public key encryption. In Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda, editors, *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 1108–1122, 2023. doi:10.1145/3576915.3623185.

[AHPS23] Calvin Abou Haidar, Alain Passelègue, and Damien Stehlé. Efficient updatable public-key encryption from lattices. In Ron Steinfeld Jian Guo, editor, *International Conference on the Theory and Application of Cryptology and Information Security*, pages 342–373. Springer, 2023. doi:10.1007/978-981-99-8733-7_11.

[AKSY22] Shweta Agrawal, Elena Kirshanova, Damien Stehlé, and Anshu Yadav. Practical, round-optimal lattice-based blind signatures. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 39–53, 2022. doi:10.1145/3548606.3560650.

[ALS16] Shweta Agrawal, Benoît Libert, and Damien Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In Jonathan Katz Matthew Robshaw, editor, *Annual International Cryptology Conference*, pages 333–362. Springer, 2016. doi:10.1007/978-3-662-53015-3_12.

[ASP12] Jacob Alperin-Sheriff and Chris Peikert. Circular and kdm security for identity-based encryption. In Mark Manulis Marc Fischlin, Johannes Buchmann, editor, *International Workshop on Public Key Cryptography*, pages 334–352. Springer, 2012. doi:10.1007/978-3-642-30057-8_20.

[BBKS07] Mihir Bellare, Alexandra Boldyreva, Kaoru Kurosawa, and Jessica Staddon. Multirecipient encryption schemes: How to save on bandwidth and computation without sacrificing security. *IEEE Transactions on Information Theory*, 53(11):3927–3943, 2007. doi:10.1109/TIT.2007.907471.

[BBS03] Mihir Bellare, Alexandra Boldyreva, and Jessica Staddon. Multi-recipient encryption schemes: Security notions and randomness re-use. In Yvo G. Desmedt, editor, *International Workshop on Public Key Cryptography*, pages 85–99, 2003. doi:10.1007/3-540-36288-6_7.

[BJTW26] Katharina Boudgoust, Corentin Jeudy, Erkan Tairi, and Weiqiang Wen. Hardness of m-lwe with general distributions and applications to leaky variants. In Edoardo Persichetti Shi Bai, editor, *IACR International Conference on Public-Key Cryptography*. Springer, 2026. (to appear). URL: https://eprint.iacr.org/2025/1472.

[DKL+23] Nico Döttling, Dimitris Kolonelos, Russell WF Lai, Chuanwei Lin, Giulio Malavolta, and Ahmadreza Rahimi. Efficient laconic cryptography from learning with errors. In Martijn Stam Carmit Hazay, editor, *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 417–446. Springer, 2023. doi:10.1007/978-3-031-30620-4_14.

[ENP24] Thomas Espitau, Guilhem Niot, and Thomas Prest. Flood and submerse: Distributed key generation and robust threshold signature from lattices. In Douglas Stebila Leonid Reyzin, editor, *Annual International Cryptology Conference*, pages 425–458. Springer, 2024. doi:10.1007/978-3-031-68394-7_14.

[ESTX24] Muhammed F Esgin, Ron Steinfeld, Erkan Tairi, and Jie Xu. Leopard: Towards practical post-quantum oblivious prfs via interactive lattice problems. *Cryptology ePrint Archive*, 2024. URL: https://eprint.iacr.org/2024/1615.

[KKPP20] Shuichi Katsumata, Kris Kwiatkowski, Federico Pintore, and Thomas Prest. Scalable ciphertext compression techniques for post-quantum kems and their applications. In Huaxiong Wang Shiho Moriai, editor, *International Conference on the Theory and Application of Cryptology and Information Security*, pages 289–320. Springer, 2020. doi:10.1007/978-3-030-64837-4_10.

[KLSS23] Duhyeong Kim, Dongwon Lee, Jinyeong Seo, and Yongsoo Song. Toward practical lattice-based proof of knowledge from hint-mlwe. In Anna Lysyanskaya Helena Handschuh, editor, *Annual International Cryptology Conference*, pages 549–580. Springer, 2023. doi:10.1007/978-3-031-38554-4_18.

[Kur02] Kaoru Kurosawa. Multi-recipient public-key encryption with shortened ciphertext. In Pascal Paillier David Naccache, editor, *International Workshop on Public Key Cryptography*, pages 48–63, 2002. doi:10.1007/3-540-45664-3_4.

[LSTW25] Zeyu Liu, Katerina Sotiraki, Eran Tromer, and Yunhao Wang. Lattice-based multi-message multi-recipient kem/pke with malicious security. In Bo-Yin Yang Goichiro Hanaoka, editor, *International Conference on the Theory and Application of Cryptology and Information Security*, pages 428–460. Springer, 2025. doi:10.1007/978-981-95-5099-9_14.

[LSW25] Russell Lai, Monisha Swarnakar, and Ivy Woo. Leaky lwe: learning with errors with semi-adaptive secret-and error-leakage. *IACR Communications in Cryptology*, pages 1–23, 2025. doi:10.62056/ah89ksuc2.

[PPS14] Alexandre Pinto, Bertram Poettering, and Jacob CN Schuldt. Multi-recipient encryption, revisited. In Kouichi Sakurai Shiho Moriai, Trent Jaeger, editor, *Proceedings of the 9th ACM symposium on Information, computer and communications security*, pages 229–238, 2014. doi:10.1145/2590296.2590329.

[WLL24] Zhedong Wang, Qiqi Lai, and Feng-Hao Liu. Ring/module learning with errors under linear leakage–hardness and applications. In Vanessa Teague Qiang Tang, editor, *IACR International Conference on Public-Key Cryptography*, pages 275–304. Springer, 2024. doi:10.1007/978-3-031-57722-2_9.

[WSS+26] Hongxiao Wang, Ron Steinfeld, Markku-Juhani O. Saarinen, Muhammed F. Esgin, and Siu-Ming Yiu. mmCipher: Batching post-quantum public key encryption made bandwidth-optimal. In Elissa Redmiles and Ben Stock, editors, *35th USENIX security symposium (USENIX Security 26)*, page (to appear), 2026. URL: https://eprint.iacr.org/2025/1000.

[WW23] Hoeteck Wee and David J Wu. Succinct vector, polynomial, and functional commitments from lattices. In Martijn Stam Carmit Hazay, editor, *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 385–416. Springer, 2023. doi:10.1007/978-3-031-30620-4_13.