IACR Communications in Cryptology
ISSN 3006-5496, Vol. 1, No. 3, 19 pages.

https://doi.org/10.62056/a3n5qj888

The Perils of Limited Key Reuse: Adaptive and Parallel Mismatch Attacks with Post-processing Against Kyber

Qian Guo[1], Erik Mårtensson[1,2,3] and Adrian Åström[4]

[1] Department of Electrical and Information Technology, Lund University, Lund, Sweden
[2] Selmer Center, Department of Informatics, University of Bergen, Bergen, Norway
[3] Advenica AB, Malmö, Sweden
[4] Lund University, Lund, Sweden

Abstract.
The Module Learning With Errors (MLWE)-based Key Encapsulation Mechanism (KEM) Kyber is NIST's new standard scheme for post-quantum encryption. As a building block, Kyber uses a Chosen Plaintext Attack (CPA)-secure Public Key Encryption (PKE) scheme, referred to as Kyber.CPAPKE. In this paper we study the robustness of Kyber.CPAPKE against key mismatch attacks.
We demonstrate that Kyber's security levels can be compromised if having access to a few mismatch queries of Kyber.CPAPKE, by striking a balance between the parallelization level and the cost of lattice reduction for post-processing. This highlights the imperative need to strictly prohibit key reuse in Kyber.CPAPKE.

We further propose an adaptive method to enhance parallel mismatch attacks, initially proposed by Shao et al. at AsiaCCS 2024, thereby significantly reducing query complexity. This method combines the adaptive attack with post-processing via lattice reduction to retrieve the final secret key entries. Our method proves its efficacy by reducing query complexity by 14.6 % for Kyber512 and 7.5 % for Kyber768/Kyber1024.

Furthermore, this approach has the potential to improve multi-value Plaintext-Checking (PC) oracle-based side-channel attacks and fault-injection attacks against Kyber itself.

Keywords: Lattice-based cryptography · Mismatch attacks · Kyber · Post-quantum standardization · KEM.

1 Introduction

The rapid development of quantum computing has significantly heightened the urgency to evolve cryptographic standards that can withstand new quantum threats. Recognizing this, the National Institute of Standards and Technology (NIST) initiated a standardization process in 2016 to foster the development of post-quantum cryptography (PQC). Among the various branches of PQC, lattice-based cryptography [AD97, Reg05] stands out for its efficiency and strong provable security. This branch has led to the selection of the Module Learning With Errors (MLWE)-based Key Encapsulation Mechanism (KEM) Kyber [SAB+20] for standardization, highlighting its prominence in the field. NIST's standardized version of the scheme is now known as Module-Lattice-based Key-Encapsulation Mechanism Standard (ML-KEM) [Nat23].

E-mail: qian.guo@eit.lth.se (Qian Guo), erik.martensson@eit.lth.se (Erik Mårtensson), adrian.astrom@outlook.com (Adrian Åström)

Received: 2024-07-08 Accepted: 2024-09-02

The majority of post-quantum KEMs that are resistant to chosen-ciphertext attacks (CCA) originate from public key encryption (PKE) schemes that are secure against chosen-plaintext attacks (CPA). These schemes are subsequently enhanced to achieve CCA security through transformations such as the Fujisaki-Okamoto (FO) method [FO99]. A growing trend (e.g., [HDV22, JMZ23, DGK24, ZJZ24]) in post-quantum KEM research involves adopting CPA-secure schemes without the FO transformation for ephemeral-key settings, tailored for protocols such as TLS 1.3, to enhance efficiency. However, before these schemes are practically deployed, it is crucial to conduct comprehensive security assessments.

A particularly relevant attack type to the CPA-secure KEMs without CCA security is keypair-reuse attacks. In 2016, Fluhrer initiated key-reuse attacks against lattice-based encryption [Flu16]. Later, Ding, Fluhrer, and Saraswathy extended these attacks to lattice-based key exchange and introduced the concept of a key mismatch attack [DFR18]. In a key mismatch attack, one communicating party's public key is reused. An adversary impersonates the other party, sends maliciously formed responses and recovers the secret key bit by bit, by repeatedly verifying if the two derived shared keys match. This type of attack can be applied to many lattice-based KEMs, with subsequent improvements in query complexities reported in various studies [BBLP18, BGRR19, BDH$^+$19, QCD19, OWT20, GMR20, HV20, QCZ$^+$21, GM23].

Inspired by a recent work on a multi-positional key mismatch attack [GM23] and recent developments in multi-value PC (Plaintext-Checking) oracle based side-channel attacks [TUX$^+$23, RRD$^+$23], Shao et al. developed techniques for conducting mismatch attacks against multiple key coefficients in parallel [SLZ24], significantly reducing the required number of queries by recovering a bit of information about multiple coefficients at once, in a single query.

The investigation of security regarding key mismatch attacks is of significant practical interest, particularly concerning the potential commonality of key-pair reuse. Notably, in crucial internet protocols such as TLS 1.3, static public keys are used in certain modes, increasing the likelihood of programming errors that lead to the reuse of ephemeral key pairs. Furthermore, the results in current research [SLZ24] suggest that a moderate level of key pair reuse–e.g., fewer than 40 times for Kyber512–might still be acceptable in post-quantum KEMs, potentially allowing real-world implementations to intentionally permit some degree of key reuse for efficiency reasons.

This study concentrates on the Kyber.CPAPKE building block of the MLWE-based KEM Kyber (which as a whole has been chosen by NIST for standardization), to assess its robustness against key mismatch attacks. We analyze how key reuse of Kyber.CPAPKE affects the concrete security of Kyber and investigate whether a limited amount of reuse can be considered secure under practical deployment scenarios.

1.1 Contributions

The primary contributions of this paper are as follows:

1. Firstly, we observe that the level of parallelization p in a parallel key mismatch attack from [SLZ24] is limited by the adversary's computational capabilities. Consequently, allowing substantial computational resources for post-processing – such as lattice reduction – can improve the performance of the attack. To minimize the required number of queries, we demonstrate how to balance the parallelization level p with the cost of using lattice reduction to solve for the remaining parts of the secret. This balance creates a new curve of query vs. computational complexity, which we illustrate in Figure 4. Our application of this optimization method shows that just two mismatch queries of Kyber.CPAPKE can compromise the claimed security levels of all three versions of Kyber. Importantly, the security of the system declines

sharply with the onset of more key reuse, highlighting the necessity for its strict prohibition.

2. Secondly, we introduce a novel methodology that leverages an adaptive approach, for improving the parallel mismatch attacks of Shao et al. [SLZ24]. This approach significantly improves query complexity compared to the original work. The improvement stems from combining the adaptive attack with post-processing via lattice reduction to recover the final secret key entries. We demonstrate the effectiveness of our approach by achieving a 14.6% and 7.5% reduction in query complexity against the Kyber.CPAPKE of Kyber512 and Kyber768/Kyber1024, respectively. Such improvement has been verified through an implementation[1]. One key limitation of the work of [SLZ24] is the inability to be adapted to all possible parallelization levels of p. We address this by strategically reserving a few positions within each block for post-processing. This simple modification allows for efficient attacks with any chosen parallelization level.

3. Finally, we investigate further applications of our proposed method. In particular, this method can significantly enhance multi-value PC-oracle-based side-channel attacks targeting the CCA-secure variant of the Kyber KEM. Additionally, it highlights the critical impact of fault-injection attacks on CCA-secure Kyber KEM, where a small number of faults may substantially decrease the complexity required for full key recovery.

1.2 Organization

The rest of the paper is organized as follows. In Section 2, we present the necessary background including CPA-secure versions of Kyber, and the model of (parallel) mismatch attacks. In Section 3, we survey previous mismatch attacks on Kyber. Then we introduce our new attack methodology in Section 4, including our implementation and cost analysis of it. This is followed by a discussion on the implications of our findings on side-channel and fault-injection attacks in Section 5, plus a couple of more suggestions on how to improve our attack. Finally, we conclude the paper and suggest future research directions in Section 6.

2 Background

Let us introduce a CPA-secure instantiation of Kyber. Note that in the official documents of Kyber, the CPA-secure versions are limited to ephemeral keys, but this constraint might be ignored in practice. To assess their key reuse resilience we create these CPA-secure instantiations. Our notations and terminology are similar to previous work on mismatch attacks, such as [QCZ+21].

- We let $\mathbf{x}||\mathbf{y}$ be the concatenation of two strings $\mathbf{x}$ and $\mathbf{y}$.
- Let $\mathbf{H}(\cdot)$ be a hash function.
- Let $\leftarrow_{\$}$ denote sampling from a distribution.
- The transpose of the matrix $\mathbf{A}$ we denote by $\mathbf{A}^{\mathsf{tr}}$.
- The central binomial distribution whose output is computed as $\sum_{i=1}^{\eta}(a_i - b_i)$, where a_i and b_i are independently and uniformly randomly sampled from $\{0, 1\}$, we denote by $\mathbf{B}_\eta$.

[1] Available at https://github.com/AdrianAstrm/Adaptive-and-Parallel-Key-Mismatch-Attack-on-Kyber.

Alice		**Bob**
1. Generate matrix $\mathbf{a} \in \mathcal{R}_q^{l\times l}$		
$\mathbf{s}_A, \mathbf{e}_A \leftarrow_\$ \mathbf{B}_\eta^l$		2. $\mathbf{m} \leftarrow_\$ \{0,1\}^{256}$
$\mathbf{P}_A \leftarrow \mathbf{a} \circ \mathbf{s}_A + \mathbf{e}_A$		Generate matrix $\mathbf{a} \in \mathcal{R}_q^{l\times l}$
Output: $(\mathbf{s}_A, \mathbf{P}_A)$	$\xrightarrow{\mathbf{P}_A}$	$\mathbf{s}_B \leftarrow_\$ \mathbf{B}_\eta^l, \mathbf{e}_B \leftarrow_\$ \mathbf{B}_{\eta'}^l, \mathbf{e}'_B \leftarrow_\$ \mathbf{B}_{\eta'}$
		$\mathbf{P}_B \leftarrow \mathbf{a} \circ \mathbf{s}_B + \mathbf{e}_B$
		$\mathbf{v}_B \leftarrow \mathbf{P}_A^{\mathrm{tr}} \circ \mathbf{s}_B + \mathbf{e}'_B$
		$+ \mathbf{Decompress}_q(\mathbf{m}, 2)$
3. $\mathbf{u}_A \leftarrow \mathbf{Decompress}_q(\mathbf{c}_1, 2^{d_{\mathbf{P}_B}})$		$\mathbf{c}_1 \leftarrow \mathbf{Compress}_q(\mathbf{P}_B, 2^{d_{\mathbf{P}_B}})$
$\mathbf{v}_A \leftarrow \mathbf{Decompress}_q(\mathbf{c}_2, 2^{d_{\mathbf{v}_B}})$	$\xleftarrow{\mathbf{P}_B, \mathbf{c}_1, \mathbf{c}_2}$	$\mathbf{c}_2 \leftarrow \mathbf{Compress}_q(\mathbf{v}_B, 2^{d_{\mathbf{v}_B}})$
$\mathbf{m}' \leftarrow \mathbf{Compress}_q(\mathbf{v}_A - \mathbf{s}_A^{\mathrm{tr}} \circ \mathbf{u}_A, 2)$		$K_B \leftarrow \mathbf{H}(\mathbf{m} \vert\vert (\mathbf{P}_B, (\mathbf{c}_1, \mathbf{c}_2)))$
$K_A \leftarrow \mathbf{H}(\mathbf{m}' \vert\vert (\mathbf{P}_B, (\mathbf{c}_1, \mathbf{c}_2)))$		

Figure 1: The CPA-secure version of Kyber.

2.1 CPA-Secure Version of Kyber

Kyber [SAB+20] is the KEM part of CRYSTALS (Cryptographic Suite for Algebraic Lattices), based on the Module Learning with Errors (MLWE) problem. In the fourth round NIST selected Kyber as their scheme for PKE/KEM. Just like in the work of [QCZ+21], we describe a potential instantiation of a CPA-secure version of Kyber KEM in Figure 1 by invoking the functions of Kyber.CPAPKE from [SAB+20].

By $\mathcal{R}_q$ we denote the polynomial ring $\mathbb{Z}_q[x]/(x^n + 1)$, for $q = 3329$ and $n = 256$. Let $\circ$ ($+$ or $-$) be the corresponding multiplication (addition or subtraction) in the ring. Let l denote the rank of the module, which is set to be $2, 3,$ and 4, for the three different versions, Kyber512, Kyber768 and Kyber1024 respectively. By calling a pseudorandom function from a public seed Alice and Bob generate a matrix $\mathbf{a}$. Kyber employs two central binomial distributions $\mathbf{B}_\eta$ and $\mathbf{B}_{\eta'}$, as shown in Figure 1. Kyber512 uses $(\eta, \eta') = (3, 2)$ and both Kyber768 and Kyber1024 use $(\eta, \eta') = (2, 2)$. Kyber512 and Kyber768 use $(d_{\mathbf{P}_B}, d_{\mathbf{v}_B}) = (10, 4)$, while Kyber1024 uses $(d_{\mathbf{P}_B}, d_{\mathbf{v}_B}) = (11, 5)$. The $\mathbf{Compress}_q(x, p)$ function maps x from module q to module p by computing

$$\mathbf{Compress}_q(x, p) = \lceil x \cdot p/q \rfloor \mod {}^+p,$$

where $r' = r \mod {}^+p$ represents the unique element r' in the range $-\frac{p}{2} < r' \leq \frac{p}{2}$ such that $r' \equiv r \pmod{p}$. Its inverse function is defined as

$$\mathbf{Decompress}_q(x, p) = \lceil x \cdot q/p \rfloor.$$

When applying $\mathbf{Compress}_q(x, p)$ or $\mathbf{Decompress}_q(x, p)$ to a vector/polynomial, then we compute the output coefficient by coefficient.

2.2 The Threat Model – Parallel Mismatch Attacks

This work focuses on the key mismatch threat model against an ephemeral-only KEM, which reuses the keypair. Alice reuses her keypair $(\mathbf{s}_A, \mathbf{P}_A)$. The adversary Eve takes advantage of this by impersonating Bob to recover Alice's secret key $\mathbf{s}_A$ by communicating with Alice. We build an oracle to simulate the decapsulation of Alice with input including the pair $(\mathbf{P}_B, \mathbf{c})$ chosen by Eve and the corresponding shared key K_B. We let $(\mathbf{c}_1, \mathbf{c}_2)$ be denoted by $\mathbf{c}$. The oracle $\mathcal{O}$ calls Alice's decapsulation function and obtains the shared key K_A. It outputs 1 if the shared keys K_A and K_B match and 0 otherwise. The goal of a mismatch attack is to recover Alice's key by selecting the chosen pairs of the form $(\mathbf{P}_B, \mathbf{c})$ and iteratively querying the oracle $\mathcal{O}$. In a parallel mismatch attack Eve enumerates multiple keys K_B, learning multiple bits of information about $\mathbf{s}_A$ by observing which key matches Alice's key K_A.

3 Mismatch Attacks

In this section we cover previous work on mismatch attacks against Kyber. We give fairly detailed descriptions of how to choose the parameters for the different approaches, to make it easier to understand our suggested improved algorithm in Section 4. Throughout the whole section we explain how to perform the attacks on Kyber1024. Attacks against other versions of Kyber simply require changing a few parameter values.

In a mismatch attack, Eve impersonates Bob and recovers Alice's secret key $\mathbf{s}_A$ step by step. Now consider Figure 1. Alice computes $\mathbf{m}'$ purely as a function of $(\mathbf{P}_B, \mathbf{c})$. We see that the keys K_A and K_B match if and only if Alice's computed message $\mathbf{m}'$ matches the message $\mathbf{m}$ that Eve chooses. Thus, Eve maliciously sets the parameters $(\mathbf{P}_B, \mathbf{c})$ and $\mathbf{m}$ such that the output of the oracle tells her something about the secret $\mathbf{s}_A$. In other words, whether or not K_A and K_B match tells Eve something about the values in $\mathbf{s}_A$.

3.1 One-Positional Mismatch Attacks

The simplest works on mismatch attacks recover one position at a time. Let us explain in some detail how this works. We focus on the position with index 0. When the subscript A is understood we let s_i denote $\mathbf{s}_A[i]$.

Eve chooses the message $\mathbf{m} = [1, 0, \ldots, 0]$. She sets $\mathbf{P}_B = [\lceil \frac{q}{32} \rfloor, 0, \ldots, 0]$. She computes $\mathbf{c}_1 = \mathbf{Compress}_q(\mathbf{P}_B, 2^{d_{\mathbf{P}_B}})$ and lets $\mathbf{c}_2 = [h, 0, \ldots, 0]$. Here h is a parameter that is adjusted depending on what information about the secret Eve wants to extract. Alice calculates $\mathbf{u}_A = \mathbf{Decompress}_q(\mathbf{c}_1, 2^{d_{\mathbf{P}_B}}) = \mathbf{P}_B$ and $\mathbf{v}_A = \mathbf{Decompress}_q(\mathbf{c}_2, 2^{d_{\mathbf{v}_B}}) = [\lceil \frac{q}{32} h \rfloor,\ 0,\ \ldots,\ 0]$. Finally, Alice gets

$$\begin{aligned} \mathbf{m}'[0] = & \mathbf{Compress}_q((\mathbf{v}_A - \mathbf{s}_A^{\mathsf{tr}} \mathbf{u}_A)[0], 2) && (1)\\ = & \mathbf{Compress}_q(\mathbf{v}_A[0] - (\mathbf{s}_A^{\mathsf{tr}} \mathbf{u}_A)[0], 2) && (2)\\ = & \left\lceil \frac{2}{q} \left(\left\lceil \frac{q}{32} h \right\rfloor - \mathbf{s}_A[0] \left\lceil \frac{q}{32} \right\rfloor \right) \right\rfloor \mod 2. && (3) \end{aligned}$$

Now given a split of the possible values of s_i into any possible two adjacent intervals. It can be shown that by adjusting h, the value of $\mathbf{m}'[0]$ can teach Eve which of these two intervals s_0 belongs to. Let us now show why Alice's received message is equal to 0 - by construction - on all positions with non-zero index. Because $\mathbf{v}_A[i] = 0$, for all indexes $i \neq 0$, for all these indexes the value of $\mathbf{m}'$ simplifies to

$$\begin{aligned} \mathbf{m}'[i] = & \mathbf{Compress}_q((\mathbf{v}_A - \mathbf{s}_A^{\mathsf{tr}} \mathbf{u}_A)[i], 2) && (4)\\ = & \mathbf{Compress}_q(\mathbf{v}_A[i] - (\mathbf{s}_A^{\mathsf{tr}} \mathbf{u}_A)[i], 2) && (5)\\ = & \left\lceil \frac{2}{q} \left(-\mathbf{s}_A[i] \left\lceil \frac{q}{32} \right\rfloor \right) \right\rfloor \mod 2. && (6) \end{aligned}$$

Before applying the outer rounding the expression is bounded in absolute value by $2/3329 \cdot 2 \cdot 105 = 0.126\ldots < 1/2$. The value is thus always equal to 0 after being rounded to the nearest integer. Therefore $\mathbf{m}'[i] = 0$, for $i \neq 0$.

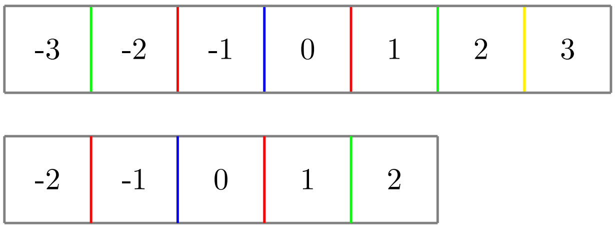

Figure 2: Illustrations of the mismatch attacks on all versions of Kyber from [QCZ+21]. The figure is a slight modification of Figure 3 from [GM23]. The bottom half of the figure covers Kyber768 and Kyber1024, while the top half covers Kyber512.

For details on how to modify the attack to recover $\mathbf{s}_A[i]$, for $i \neq 0$, see for example [GM23]. In [QCZ+21] the authors derived the so called Hamming bound for this type of attack. This bound corresponds to the best possible if queries splitting the possible values of $\mathbf{s}_A[i]$ into any possible two subsets are available. This bound is reached by using Huffman coding.

They also showed how to select the parameter h in each step to get close to the Hamming bound. See Figure 2 for an illustration. Each colored line corresponds to a query, splitting the remaining possible secret values into two adjacent intervals. The blue/red/green/yellow line corresponds to the first/second/third/fourth query needed to recover the secret value. Given that the secret values are sampled from a central binomial distribution the attack needs an expected

$$2\left(\frac{1}{16}+\frac{4}{16}+\frac{6}{16}\right)+3\left(\frac{1}{16}+\frac{4}{16}\right)=\frac{37}{16}=2.3125$$

queries to recover one coefficient for Kyber768/Kyber1024 and

$$2\left(\frac{15}{64}+\frac{20}{64}\right)+3\left(\frac{1}{64}+\frac{6}{64}+\frac{15}{64}\right)+4\left(\frac{1}{64}+\frac{6}{64}\right)=\frac{164}{64}=2.5625$$

queries to recover one coefficient for Kyber512, respectively. In [GM23] it was shown that for one-positional mismatch attacks against Kyber, the attack of [QCZ+21] is (most likely) optimal.

3.2 Multi-Positional Mismatch Attacks

In [GM23] the authors remove the constraint of recovering only one coefficient at a time and thereby break the Hamming bound of [QCZ+21]. Let us explain their idea for attacking two positions at a time.

3.2.1 Two-Positional Mismatch Attacks on Kyber

We will show how to obtain s_0 and s_{128}. Eve lets $\mathbf{m}$ be equal to 0 on all positions, except that $\mathbf{m}[0] = 1$ and/or $\mathbf{m}[128] = 1$. She lets $\mathbf{P}_B$ be 0 on all positions, except that $\mathbf{P}_B[0] = b_1 \cdot \lceil \frac{q}{32} \rfloor$ and $\mathbf{P}_B[128] = b_2 \lceil \frac{q}{32} \rfloor$, for $b_1, b_2 \in \{-1, 0, 1\}$. Also, she sets $\mathbf{c}_2$ to 0 on all positions, except that $\mathbf{c}_2[0] = h_1$ and $\mathbf{c}_2[128] = h_2$.[2] Next, let us compute $\mathbf{m}'[0]$ and $\mathbf{m}'[128]$.

[2] To retrieve the positions s_i and s_{128+i}, where $1 \leq i \leq 127$, we can for example make the following adjustments. Let $\mathbf{m}$ be equal to 0 on all positions except that $\mathbf{m}[i] = 1$ and/or $\mathbf{m}[128+i] = 1$. Also, let $\mathbf{c}_2$ be 0 on all positions, except that $\mathbf{c}_2[i] = h_1$ and $\mathbf{c}_2[128+i] = h_2$.

$$\begin{aligned}\mathbf{m}'[0] =& \mathbf{Compress}_q(\mathbf{v}_A[0] - (\mathbf{s}_A^{\mathsf{tr}}\mathbf{u}_A)[0], 2) & (7)\\ =& \left\lceil \frac{2}{q}\left(\left\lceil \frac{q}{32}h_1 \right\rfloor - \left(\mathbf{s}_A[0]b_1\left\lceil \frac{q}{32}\right\rfloor - \mathbf{s}_A[128]b_2\left\lceil \frac{q}{32}\right\rfloor\right)\right)\right\rfloor \mod 2, & (8)\end{aligned}$$

$$\begin{aligned}\mathbf{m}'[128] =& \mathbf{Compress}_q(\mathbf{v}_A[128] - (\mathbf{s}_A^{\mathsf{tr}}\mathbf{u}_A)[128], 2) & (9)\\ =& \left\lceil \frac{2}{q}\left(\left\lceil \frac{q}{32}h_2 \right\rfloor - \left(\mathbf{s}_A[0]b_2\left\lceil \frac{q}{32}\right\rfloor + \mathbf{s}_A[128]b_1\left\lceil \frac{q}{32}\right\rfloor\right)\right)\right\rfloor \mod 2. & (10)\end{aligned}$$

For an integer i, with $1 \le i \le 127$ we get

$$\begin{aligned}\mathbf{m}'[i] =& \mathbf{Compress}_q(-(\mathbf{s}_A^{\mathsf{tr}}\mathbf{u}_A)[i], 2) & (11)\\ =& \left\lceil \frac{2}{q}\left(-\left(\mathbf{s}_A[i]b_1\left\lceil \frac{q}{32}\right\rfloor - \mathbf{s}_A[128+i]b_2\left\lceil \frac{q}{32}\right\rfloor\right)\right)\right\rfloor \mod 2, & (12)\end{aligned}$$

$$\begin{aligned}\mathbf{m}'[128+i] =& \mathbf{Compress}_q(-(\mathbf{s}_A^{\mathsf{tr}}\mathbf{u}_A)[128+i], 2) & (13)\\ =& \left\lceil \frac{2}{q}\left(-\left(\mathbf{s}_A[i]b_2\left\lceil \frac{q}{32}\right\rfloor + \mathbf{s}_A[128+i]b_1\left\lceil \frac{q}{32}\right\rfloor\right)\right)\right\rfloor \mod 2. & (14)\end{aligned}$$

For both of these positions the expression within the outer rounding function is bounded in absolute value by $2/3329 \cdot 2 \cdot 2 \cdot 105 = 0.252\ldots < 1/2$. Hence these values are always rounded to 0 and thus $\mathbf{m}'[i] = 0$, for $i \neq 0, 128$.

Let a two-dimensional grid represent all possible combinations of values that (s_0, s_{128}) can take. The authors of [GM23] go into great details on geometrical interpretations of how to interpret different possible splits you can make in two dimensions, depending on how you set the parameters.

The one-dimensional cuts from Section 3 correspond to making horizontal or vertical (planar) cuts in this two-dimensional grid. You can also make triangular-shaped cuts originating from any of the corners of the grid. Finally, by combining two planar cuts or two triangular cuts, you can perform rectangular cuts or intersecting triangular cuts respectively. In [GM23] the authors show how to optimize mismatch attacks using these types of splits. As our improvement in Section 4 essentially does one-dimensional cuts, but in parallel, we refer to [GM23] for further details on two-dimensional cuts.

3.2.2 Hyperrectangular Cuts

In [GM23] the authors also showed how to generalize the one-dimensional mismatch attacks in another way. Instead of making planar cuts in one or two dimensions at a time, they make planar cuts with respect to an arbitrary subset of the positions, at a time. Let us explain their idea. Let $I \subset \{0, 1, \ldots, n-1\}$ be the set of indexes that we want to make planar splits with respect to. Let $\mathbf{m}[i] = 1$, for $i \in I$, and $\mathbf{m}[i] = 0$, for $i \notin I$. Let $\mathbf{P}_B[0] = \lceil \frac{q}{32} \rfloor$ and let $\mathbf{P}_B$ be equal to 0 on all other positions. Let $\mathbf{c}_2[i] = h_i$, for $i \in I$ and let $\mathbf{c}_2[i] = 0$, for $i \notin I$. Here h_i are the parameters deciding the precise planar cut with respect to each dimension. For $i \in I$ we now get

$$\begin{aligned}\mathbf{m}'[i] =& \mathbf{Compress}_q(\mathbf{v}_A[i] - (\mathbf{s}_A^{\mathsf{tr}}\mathbf{u}_A)[i], 2) & (15)\\ =& \left\lceil \frac{2}{q}\left(\left\lceil \frac{q}{32}h_i \right\rfloor - \mathbf{s}_A[i]\left\lceil \frac{q}{32}\right\rfloor\right)\right\rfloor \mod 2. & (16)\end{aligned}$$

For $i \notin I$ we get

$$\begin{aligned}\mathbf{m}'[i] =& \mathbf{Compress}_q(\mathbf{v}_A[i] - (\mathbf{s}_A^{\mathbf{tr}}\mathbf{u}_A)[i], 2) && (17)\\ =& \left\lceil \frac{2}{q}\left(-\mathbf{s}_A[i]\left\lceil\frac{q}{32}\right\rfloor\right)\right\rfloor \mod 2, && (18)\end{aligned}$$

which simplifies to 0 just like in the one-dimensional mismatch attack.

3.3 Post-Processing with Lattice Reduction

Other than the information from the mismatch queries the adversary has access to LWE samples. Having recovered parts of the secret through mismatch queries, solving for the remaining parts of the secret using lattice reduction was initially studied in [GM23, MJZ22]. In both works it was showed that the number of queries needed to recover the full key was significantly reduced using this type of post-processing.

4 Adaptive Parallel Mismatch Attacks

Now let us introduce our adaptive mismatch attacks. To do so, let us first introduce parallel mismatch attacks generally in Section 4.1. There we cover the parallel mismatch attacks of Shao et al. [SLZ24], but also the parallel PC oracle attacks of [TUX+23, RRD+23], translated to the mismatch attack setting. In Section 4.2 we introduce our improved, adaptive version of parallel mismatch attacks against Kyber. Finally, we discuss our implementation of our adaptive, parallel mismatch attack in Section 4.3 and the computational cost analysis of it in Section 4.4.

4.1 Parallel Mismatch Attacks

In a recent paper [SLZ24] Shao et al. showed how to do parallelized mismatch attacks, packing what was previously p different queries into a single query. Their strategy is very similar to recent work on parallel PC oracle attacks in [TUX+23, RRD+23]. At the cost of $\mathcal{O}(2^p)$ time the authors were able to gain (up to) p bits of information at a time instead of just (up to) 1 bit. This allowed them to trivially break the Shannon bound[3] and drastically improve upon mismatch attacks on Kyber.

Their attack is somewhat similar to the hyperrectangular cuts described in Section 3.2.2. We describe it in detail for Kyber1024. Let us describe a slight generalization of the attack of [SLZ24]. Let $I \subset \{0, 1, \ldots, 127\}$ be an index set with p positions. Our attack targets positions i and $i + 128$, where $i \in I$. Eve lets $\mathbf{P}_B$ be equal to 0 on all positions, except that $\mathbf{P}_B[0] = b_1\lceil\frac{q}{32}\rfloor$ and $\mathbf{P}_B[128] = b_2\lceil\frac{q}{32}\rfloor$, where $|b_1| + |b_2| \leq 3$. Let $\mathbf{c}_2[i] = 0$, for $i \notin I$ and $\mathbf{c}_2[i] = h_i$, for $i \in I$. Finally Eve computes $\mathbf{c}_1 = \mathbf{Compress}_q(\mathbf{P}_B, 2^{d_{\mathbf{P}_B}})$ and sends $(\mathbf{P}_B, \mathbf{c}_1, \mathbf{c}_2)$ to Alice. Alice then calculates $\mathbf{u}_A = \mathbf{Decompress}_q(\mathbf{c}_1, 2^{d_{\mathbf{P}_B}}) = \mathbf{P}_B$ and $\mathbf{v}_A = \mathbf{Decompress}_q(\mathbf{c}_2, 2^{d_{\mathbf{v}_B}})$. Here, $\mathbf{v}_A[i] = \lceil\frac{q}{32}h_i\rfloor$, for $i \in I$ and $\mathbf{v}_A[i] = 0$, for $i \notin I$. Finally, Alice gets

$$\begin{aligned}\mathbf{m}'[i] =& \mathbf{Compress}_q(\mathbf{v}_A[i] - (\mathbf{s}_A^{\mathbf{tr}}\mathbf{u}_A)[i], 2) && (19)\\ =& \left\lceil \frac{2}{q}\left(\left\lceil\frac{q}{32}h_i\right\rfloor - \left(\mathbf{s}_A[i]b_1\left\lceil\frac{q}{32}\right\rfloor - \mathbf{s}_A[i+128]b_2\left\lceil\frac{q}{32}\right\rfloor\right)\right)\right\rfloor \mod 2, && (20)\end{aligned}$$

[3] That bound of course assumes that the attacker can only gain up to 1 bit of information per query. The new lower bound of the mismatch attack is the Shannon entropy divided by p, assuming that the attacker does no post-processing.

for $i \in I$, and

$$\mathbf{m}'[i] = \mathbf{Compress}_q(\mathbf{v}_A[i] - (\mathbf{s}_A^{\mathbf{tr}}\mathbf{u}_A)[i], 2) \tag{21}$$

$$= \left\lceil \frac{2}{q} \left(- \left(\mathbf{s}_A[i] b_1 \left\lceil \frac{q}{32} \right\rfloor - \mathbf{s}_A[i+128] b_2 \left\lceil \frac{q}{32} \right\rfloor \right) \right) \right\rfloor \mod 2, \tag{22}$$

for $i \notin I$. Next, Eve enumerates all the 2^p different messages of $\mathbf{m}$ that are non-zero with respect to the indexes in I, until she finds one such that $\mathbf{m}$ and $\mathbf{m}'$ match[4]. On expectation it takes Eve 2^{p-1} steps to find the matching message $\mathbf{m}$. Finding a matching message will teach Eve something about all the values $\mathbf{s}_A[i]$ and $\mathbf{s}_A[i+128]$, for $i \in I$, at the same time. In non-parallel mismatch attacks, we simply compare against a single message $\mathbf{m}$, leading to a totally insignificant computational effort. However, checking whether K_A and K_B match takes time. As we step up p, while we can attack more positions at a time, we also increase the computational cost of the attack. Thus we get a trade-off between time and mismatch queries.

Assume w.l.o.g. that $b_1 \neq 0$. If also $b_2 \neq 0$, then we make p triangular cuts in parallel. If $b_2 = 0$, then we make p planar cuts in parallel. Note that for all the p parallel cuts, either all of them must be planar or all of them must be triangular.

Notice that when doing parallel mismatch attacks, it makes no sense to try rectangular cuts in parallel or intersecting triangular cuts in parallel[5]. This can also be seen in [SLZ24] where the authors only make planar or triangular cuts in each step[6].

4.1.1 Parameter Selection Strategy

The strategy of [TUX+23, RRD+23] - translated to our setting - is to for each attacked position make one-dimensional queries to minimize the number of queries needed to recover the least likely secret value. Among the strategies that achieve this minimal number, they choose the strategy that minimizes the expected number of queries. Notice that while they do not describe their strategy applied to mismatch attacks, their strategy can be applied to mismatch attacks. For Kyber768/Kyber1024 their attacks correspond to those of [QCZ+21], but in parallel. For Kyber512, they modify the attack to guarantee recovering the secret in 3 queries. Note that while the strategy of [QCZ+21] is faster on expectation for attacking one position at a time, it is slower when attacking many positions in parallel. If p is large, then it is highly likely that at least one of the p secret values is equal to -3 or 3, making the attack strategy of [QCZ+21] take 4 queries to fully recover all p values[7].

The authors of [SLZ24] do parallel mismatch attacks where they attack p pairs of positions in parallel, instead of just p positions. For Kyber512 they devise a strategy that always recovers p pairs in 6 queries. This matches the performance of [TUX+23, RRD+23]. For Kyber768/Kyber1024 their corresponding strategy recovers p pairs in 5 queries. This improves upon the strategy of [TUX+23, RRD+23] and is considered the state-of-the-art for parallel mismatch attacks against Kyber768/Kyber1024. In either case, this process is repeated

$$\left\lceil \frac{256}{2p} \right\rceil \tag{23}$$

[4] Which in turn is tested by noting that K_A and K_B match.

[5] Both of these types of queries correspond to making two cuts and getting the answer to the AND of the results. If the keys match, then we get a YES answer for both queries. If the keys do not match, then we do not learn which of the two answers correspond to a NO. For the parallel mismatch attacks we need to match the answers with respect to all p parallel queries.

[6] Even though they do not explain why they do not do the other types of splits.

[7] Unlike our adaptive attacks, described in Section 4.2, they perform queries until all p values are fully recovered. Thus, in their attack it is not possible to take advantage of recovering some of the positions in less than 3 queries.

Table 1: The expected number of queries needed to recover $2p$ positions using different parallel mismatch attack strategies.

	Kyber512	Kyber768/Kyber1024
Non-adaptive single [TUX+23, RRD+23]	6	6
Non-adaptive pairwise [SLZ24]	6	5
Adaptive single (this paper)	5.125	4.625

times to recover a full block of 256 positions. Notice that for fairly large values of p, like $p = 32$, this starts to be a limitation. The smallest value larger than 32 that decreases the expression in (23) is $p = 64$, which leads to a very drastic increase in computational effort.

To explain their strategy of making the performance optimal for the worst-case pairs, let us compare against the strategy for Kyber768/Kyber1024, from Figure 10 of [GM23]. On expectation, only around 4.1 queries are needed to recover a secret key pair. However, for the least likely key pairs 7 queries are needed. For somewhat large values of p, like $p = 32$, the limiting performance factor of a pure mismatch attack is the number of queries needed to recover the least likely secret key pairs, not the expected number to recover a secret key pair.

Notice that in all three works [TUX+23, RRD+23, SLZ24] the attacks are non-adaptive. In [TUX+23, RRD+23] the attack starts recovering new positions first when the current p positions are all fully recovered. The same is true for the p pairs in the attack of [SLZ24].

A trivial way of working around the problem in (23) of fine-tuning p to fit the computational resources is to solve a few positions using post-processing with lattice reduction. This way we can make use of having computational resources slightly larger than what is needed to let $p = 32$. If we for example leave 25 positions per 256 positions block, then we can let $p = 33$. We do not need to increase p all the way up to 64 to improve.

4.2 Our Adaptive Parallel Mismatch Attacks

Let us now introduce our improved, adaptive version of parallel mismatch attacks. Our main improvement over [SLZ24] is to revisit the mismatch attack of [QCZ+21], but to do it in parallel in a more efficient way. We let $I = \{0, 1, \ldots, p-1\}$ and let $b_2 = 0$. Thus, we decide to perform planar cuts in parallel. Instead of performing queries until everyone of the entries with indexes in I is recovered, we work adaptively. As soon as a position is uniquely determined, we replace that position of I by the next non-solved entry in the current block.

While this strategy is slow for large p when recovering every single position[8], as long as we leave at least p positions for recovery by post-processing, the performance of it is greatly improved. If the size of each block is large compared to p, then we can model the adaptive parallel mismatch attack as p one-dimensional mismatch attacks going on in parallel. Using the performance for one position from [QCZ+21] we then need an expected 2.5625 or 2.3125 to recover p positions when attacking Kyber512 or Kyber768/Kyber1024 respectively.

Now let us summarize the expected number of queries needed to recover $2p$ positions using our work versus previous works.

Compared to the previous state-of-the-art we reduce the expected number of queries by roughly 14.6 % for Kyber512 and 7.5 % for Kyber768/Kyber1024[9].

[8] Since the performance of the strategy is ultimately limited by the number of queries needed to recover the least likely secret key values.

[9] As the computational cost of the mismatch attack is proportional to the number of queries (see Section 4.4), we actually improve slightly more than described here.

This model starts to break down for two reasons for large p, both of which have to do with the the relative size of p compared to the block size 256.

- When p is large compared to the block size 256, then we cannot reach the asymptotic performance of 2.3125/2.5625 queries per p positions.
- At the end of a block there might be less than p positions to solve for, at which point we can no longer have a parallelization level of p. As discussed already, we partly mitigate this by leaving the last positions for post-processing and moving on to attacking the next block, as soon as less than p positions of the block remain to be found.

The details of how well the model works as a function of p is covered in Section 4.3. A small additional benefit of our attack is that it allows us to use an arbitrary number of queries, instead of a multiple of 3 or 5/6 like in [TUX$^+$23, RRD$^+$23] and [SLZ24] respectively.

4.3 Implementation Results

We implemented our algorithm in C, extending the work of [QCZ$^+$21]. To better understand the precise performance of our algorithm we ran experiments using the implementation, summarized in Figure 3. For Kyber512 and Kyber1024 we plot the expected cost to recover p secret entries - using our adaptive approach, the non-adaptive approach of [TUX$^+$23, RRD$^+$23] and for Kyber1024 the pairwise approach of [SLZ24][10] - as a function of p[11]. For all algorithms we improve upon the expected performance by leaving the last positions for post-processing and moving on to the next block, as soon as less than p entries remain to be solved for.

For our adaptive approach each data point is computed as the average of the result of running the attack 100 times. For $p \leq 22$ we ran the whole attack. For $p > 22$ we "cheated" by knowing which $\mathbf{m}$ matches $\mathbf{m}'$. We ran the rest of the attack like normal, but skipped the computationally heavy part of enumerating to find the matching vector $\mathbf{m}$. That way we were able to study how the query performance of the attack for large p, without having the computational resources to perform the whole attack[12].

For Kyber512 we see that the performance of our adaptive attack roughly matches the simple theoretical value of recovering p positions in 2.5625 queries, for small values of p. Gradually the performance gets worse and gets beaten by the non-adaptive approach of [TUX$^+$23, RRD$^+$23] for $p \geq 112$. We do not care about studying the attack for $p > 128$, as it is cheaper to solve the underlying LWE problem of Kyber512 than running parallel mismatch attacks for values of p that large.

For Kyber1024, for small values of p our adaptive approach roughly matches the simple theoretical value of recovering p positions in 2.3125 queries for small values of p. For $76 \leq p \leq 128$, the pairwise approach of [SLZ24] outperforms our method. Notice however, that the pairwise approach attacks $2p$ positions at the time, making it impossible to perform it for $p > 128$. For $p > 128$ our adaptive approach performs better than or equal to the non-adaptive approach of [TUX$^+$23, RRD$^+$23] for all p. As the secret distribution is the same for Kyber768 and Kyber1024, we can omit a separate figure for the attack performance on Kyber768. Notice however, that for Kyber768 we do not care about the performance of the attack for p $>$ 192, as it is cheaper to solve the underlying LWE problem of Kyber768 than running parallel mismatch attacks for values of p that large.

[10] For Kyber512 the pairwise approach does not improve upon the non-adaptive approach of [TUX$^+$23, RRD$^+$23].

[11] As Kyber768 uses the same distribution for the secret entries as Kyber1024, we do not need to also plot results for Kyber768.

[12] At a parallelization level of 256 of course nobody has the computational resources to perform the attack once, let alone do it a hundred times.

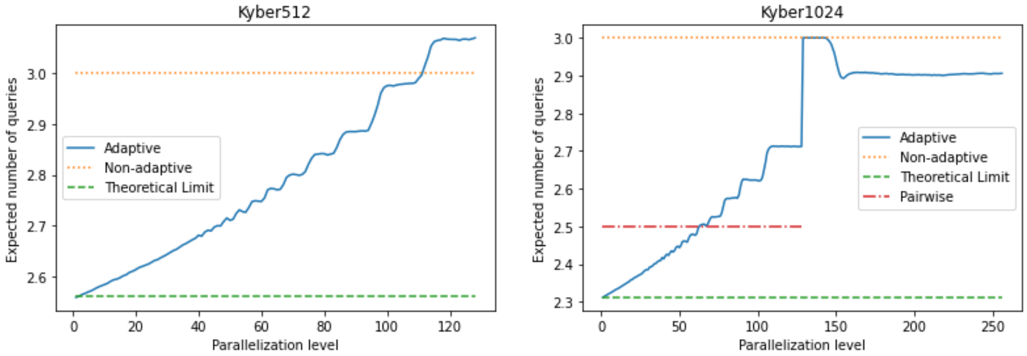

Figure 3: Expected number of queries needed to recover p positions, as a function of p, using different parallel mismatch attacks against Kyber512 and Kyber1024 respectively.

4.4 Computational Cost of Parallel Mismatch Attacks

Each time we want to test whether K_A and K_B match, we need to compute a hash value, decrypt a message and finally compare the values of two vectors. See Section 4.2 of [SLZ24] for some more details of the procedure. Here the decryption operation is the dominant part of the total time. We estimate it to be 2^{15} bit operations.

Each query corresponds to brute-forcing all possible binary keys that are zero everywhere except for indexes defined by the set I, with $|I| = p$. On expectation we need to test 2^{p-1} keys to find a matching one. Assume that we manage to recover $r = r(p, q_t)$ positions via a mismatch attack using a total of q_t queries. The number of recovered positions can be estimated as

$$r(p, q_t) \approx \left\lfloor p \frac{q_t}{r_e} \right\rceil, \tag{24}$$

where r_e corresponds to the expected number of queries needed to recover p positions with the chosen parallel mismatch attack algorithm, as estimated in Figure 3. For a given p and q_t we should of course choose the algorithm that according to Figure 3 recovers the most positions of the secret.

We need to solve for the remaining $256 \cdot l - r(p, q_t)$ positions via lattice reduction. The total cost of the mismatch attack and the post-processing becomes

$$2^{15} \cdot 2^{p-1} \cdot q_t + L(l \cdot 256 - r(p, q_t)), \tag{25}$$

where $L(n \cdot l - r(p, q_t))$ is the cost of solving the underlying LWE problem for the remaining $l \cdot 256 - r(p, q_t)$ positions not recovered from the mismatch attack. The latter cost is estimated using the Lattice-Estimator[13] [APS15].

For a given number of available queries, we should choose the parallelization level p and the mismatch strategy that minimizes the cost according to (25).

For all versions of Kyber we perform this type of optimization[14] and plot the relationship between query and bit complexity in Figure 4. Notice that for all versions of Kyber we drastically reduce the bit complexity below the security level by having access to as few as two mismatch queries[15]! In terms of reducing the bit security, there is a diminishing

[13] https://github.com/malb/lattice-estimator.

[14] The script for performing the optimization is available at https://github.com/ErikMaartensson/AdaptiveAndParallelMismatchAttack. The repository also contains a script used to generate Figure 3.

[15] As we cannot recover any positions fully with a single mismatch query, we assume that a single query does not reduce the bit security. However, partial information about p positions from a single query does actually already make the problem easier.

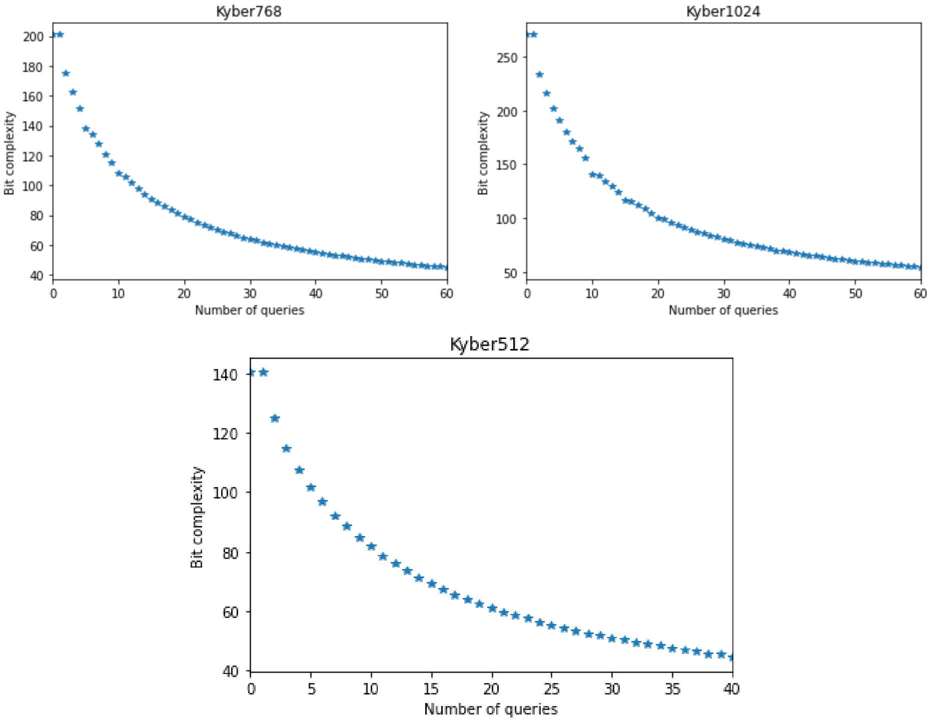

Figure 4: Bit complexity of a parallel mismatch attack with postprocessing, as a function of the number of mismatch queries.

return in terms of how much each new query simplifies the computational effort.

For higher numbers of queries, we can compare against the results of Section 6.1 of [SLZ24]. They do not attempt to optimize the total computational cost for a given number of queries. Instead, they fix different, fairly small, values for p and compute the remaining post-processing cost as a function of the number of available mismatch queries.

The authors of [SLZ24] claim that when using $p = 26$ and 78 queries, the post-processing cost is only 2^{32} when attacking Kyber1024. As their attack actually requires the number of queries to be a multiple of 5, let us study their claim according to our model. In 80 queries with $p = 26$ we recover a total of $26 \cdot 2 \cdot 80/5 = 832$ positions. According to the lattice estimator the cost of solving Kyber1024 with 192 positions remaining is roughly $2^{55.56}$. The corresponding cost for the mismatch queries is $2^{15} \cdot 2^{25} \cdot 80 = 2^{46.32}$. Thus, the total cost is still roughly $2^{55.56}$, as the post-processing cost dominates. Thus, by our calculation, when using their mismatch attack strategy, they can lower the total cost by making p a bit larger.

In comparison, we only need 58 queries to reach a total complexity cost of $2^{55.11}$. It could be that our model for the cost of post-processing is too pessimistic, but we apply it consistently to our strategy and the strategy of [SLZ24]. Thus, in an apples-to-apples comparison, our strategy is superior.

5 Discussion

In this section we discuss how our new attack methodology improves upon parallel PC oracle attacks and fault-injection attacks. We also discuss a couple of more attempts of improving our adaptive parallel mismatch attacks.

5.1 Improving Parallel PC Oracle-Based Side-Channel Attacks

PC oracle-based chosen-ciphertext side-channel attacks [GJN20, RRCB20, UXT$^+$22] against CCA-secure KEMs are a significant attack type, akin to key mismatch attacks. Both methods select ciphertexts. While the key mismatch attack focuses on the CPA-secure version, the PC oracle-based chosen-ciphertext side-channel attack targets the CCA-secure version since it can exploit side-channel information to circumvent the protection offered by CCA transformations, such as the FO transform.

The high degree of similarity between these two attack categories suggests that nearly all enhancements in key mismatch attacks can benefit PC oracle-based side-channel attacks. For instance, the one-positional key mismatch attacks introduced in [QCZ$^+$21] and the multi-positional key mismatch attacks [GM23] can enhance the query complexity of (binary) PC oracle-based side-channel attacks.

Shao et al. [SLZ24] introduced a parallel mismatch attack applicable to improving parallel or multi-value PC oracle-based side-channel attacks [TUX$^+$23, RRD$^+$23]. As we refine the work of Shao et al., the enhancements we propose in this paper can boost the efficiency of parallel or multi-value PC oracle-based side-channel attacks. The chosen ciphertexts are generated using the same method as the ciphertext selection approach described in Section 4.2.

In multi-value PC oracle-based side-channel attacks, a multi-class machine learning model is trained on side-channel information to mimic the oracle used in parallel key mismatch attacks. The number of classes required for training is 2^p, where p represents the parallelization level. However, due to constraints imposed during training, the achievable value of p is often limited. Realistic choices for p here lie in the range 8-16. As we see in Figure 3, for this range, the performance of our adaptive approach is very close to the theoretical limit covered in Table 1. Thus, we improve upon the query complexity of the state-of-the-art by roughly 14.6 % for Kyber512 and 7.5 % for Kyber768/Kyber1024 for this setting. If keeping the number of queries the same, our attack improvement instead translates to a lower total computational cost of the attack.

5.2 Implications for Fault-Injection Attacks

Fault injections in cryptographic implementations may skip critical instructions, thus potentially creating PC oracles, as evidenced by [XIU$^+$21, HPP21]. In a recent development, Mondal et al. [MKB$^+$24] proposed a technique to generate multi-value PC oracles through fault injections, specifically using a software-based approach known as RowHammer. This oracle resembles a parallel key-mismatch oracle. Our findings indicate that even a minimal number of faults can significantly compromise the security of CCA-secure KEM implementations. This aspect of our research is particularly significant given that controlling faults is typically more challenging than monitoring traces in side-channel attacks.

5.3 Other Attempts at Improving Parallel Mismatch Attacks

We made a few more attempts at improving upon the parallel mismatch attack itself, the post-processing and how to balance these two parts.

5.3.1 Triangular Splits Only

An alternative to our strategy of planar cuts in parallel is to let $b_2 \neq 0$ and use triangular splits only, as briefly mention in Section 4.1. We tried to devise a mismatch attack with this strategy. For Kyber768/Kyber1024 it performs slightly worse than the planar strategy. For Kyber512 it performs even worse. We also tried scaling down Kyber to have centered Binomial entries on $\{-1, 0, 1\}$. Here the triangular strategy performs marginally better.

In summary, it seems like the strategy of using only triangular splits is performing worse the larger set the secret entries are taken from is.

5.3.2 Synchronized Splits in Three Dimensions

The idea from [SLZ24] of making two-dimensional splits in a way to guarantee the minimum number of splits in the worst-case can be generalized to higher dimensions. Let us assume that such a splitting strategy exists and see what can be achieved for the different versions of Kyber. A triplet of entries from Kyber512 can take 7^3 different values. Since $\lceil \log_2(7^3) \rceil = \lceil \log_2(343) \rceil = 9$, this strategy recovers $3P$ entries in 9 queries. This would unfortunately be identical in performance to the strategy of [TUX$^+$23, RRD$^+$23].

For Kyber768/Kyber1024 we have $5^3 = 125$ possible triplets of entries. Since $\lceil \log_2(125) \rceil = 7$, this strategy recovers $3p$ entries in 7 queries. This would beat [SLZ24], but would be marginally worse than our adapted strategy, which requires an expected number of $3 \cdot 2.3125 = 6.9375$ queries to recover $3p$ positions.

5.3.3 Potentially More Efficient Post-processing

In a recent paper by May and Nowakowski [MN23] it was shown that having access to a surprisingly low number of so-called perfect hints about the secret vector, the LWE problem can easily be solved with LLL. The terminology originally comes from [DDGR20] and means that the attacker has access to information of the type $\mathbf{s}_A^{\mathsf{tr}}\mathbf{v}_i = l_i$, where $\mathbf{v}_i$ are known vectors and l_i are known scalars. In our setting the attacker has recovered a large number of entries of $\mathbf{s}_A$. Each such value corresponds to a hint $\mathbf{s}_A[i] = \mathbf{s}_A^{\mathsf{tr}}\mathbf{e}_i = l_i$, where $\mathbf{e}_i$ is a unit vector. In May's and Nowakowski's setting, the vectors $\mathbf{v}_i$ take uniformly random values from $\mathbb{Z}_q$.

Given the perfect hints, May and Nowakowski create a matrix where each column consists of a vector $\mathbf{v}_i$ and the corresponding value l_i. The larger the absolute value of the determinant of this matrix is, the easier the LWE problem with these hints is. While uniformly random vectors $\mathbf{v}_i$ lead to very large determinants, the hints in our setting lead to very small determinants. In our case, their approach boils down to reducing the dimension of the problem by the number of hints we are given, which matches the strategy from Section 4.4.

5.3.4 Enumeration vs. Post-processing

The key enumeration part of parallel mismatch attack is trivial to parallelize and requires negligible amounts of memory. The lattice reduction attack that the lattice estimator suggests requires an exponential amount of memory and is highly non-trivial to parallelize. Thus, it might in practice be faster to leave some more of the total number of secret entries for the mismatch attack part. As the impact of the memory requirement of lattice reduction algorithms is an active research area in itself, we consider taking this aspect into consideration when optimizing the cost out of scope for this work.

6 Conclusions and Future Work

In this paper, we highlight a critical vulnerability in a CPA-secure instantiation of Kyber, the chosen NIST post-quantum KEM, against key mismatch attacks. We demonstrate that parallelized attacks with post-processing lattice reduction can jeopardize Kyber's claimed security levels with a minimal number of queries. This finding emphasizes the strict prohibition of key reuse in a CPA-secure version of Kyber. Also, we improve upon the results in [SLZ24] by employing an adaptive strategy. By reserving a minor portion of the positions for post-processing, we manage to decrease the anticipated query count by approximately 14.6 % for Kyber512 and 7.5 % for Kyber768/Kyber1024, relative to [SLZ24]. Our advancements could likewise be utilized to further improve the parallel or multi-value PC oracle-based side-channel attacks against the CCA-secure Kyber KEM as presented in [TUX+23, RRD+23].

In considering future avenues of research, we propose a real-world evaluation of the side-channel attack enhancements enabled by our techniques (Section 5.2). This evaluation should quantify the practical improvement in attack efficacy using an experimental setup. Additionally, investigating the deployment of parallel mismatch attacks coupled with post-processing on high-performance computing resources, particularly those with extensive RAM, presents another exciting avenue for exploration. This research could provide insights into the practicality of our techniques in real-world scenarios.

Acknowledgment

QG was partially funded by the Swedish Research Council (grant numbers 2019-04166, 2021-04602, and 2023-03654), the Swedish Civil Contingencies Agency (grant number 2020-11632), the Crafoord foundation, and the Swedish Foundation for Strategic Research (Grant No. RIT17-0005). EM was funded by the project "Kvantesikker Kryptografi" from the National Security Authority of Norway. Both QG and EM were partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

[AD97] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '97, pages 284–293, New York, NY, USA, 1997. Association for Computing Machinery. doi:10.1145/258533.258604.

[APS15] Martin R Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015. doi:10.1515/jmc-2015-0016.

[BBLP18] Daniel J. Bernstein, Leon Groot Bruinderink, Tanja Lange, and Lorenz Panny. HILA5 Pindakaas: On the CCA security of lattice-based encryption with error correction. In Antoine Joux, Abderrahmane Nitaj, and Tajjeeddine Rachidi, editors, *AFRICACRYPT 18*, volume 10831 of *LNCS*, pages 203–216. Springer, Heidelberg, May 2018. doi:10.1007/978-3-319-89339-6_12.

[BDH+19] Ciprian Băetu, F. Betül Durak, Loïs Huguenin-Dumittan, Abdullah Talayhan, and Serge Vaudenay. Misuse attacks on post-quantum cryptosystems. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part II*, volume 11477 of *LNCS*, pages 747–776. Springer, Heidelberg, May 2019. doi:10.1007/978-3-030-17656-3_26.

[BGRR19] Aurélie Bauer, Henri Gilbert, Guénaël Renault, and Mélissa Rossi. Assessment of the key-reuse resilience of NewHope. In Mitsuru Matsui, editor, *CT-RSA 2019*, volume 11405 of *LNCS*, pages 272–292. Springer, Heidelberg, March 2019. doi:10.1007/978-3-030-12612-4_14.

[DDGR20] Dana Dachman-Soled, Léo Ducas, Huijing Gong, and Mélissa Rossi. LWE with side information: Attacks and concrete security estimation. In Daniele

Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 329–358. Springer, Heidelberg, August 2020. doi:10.1007/978-3-030-56880-1_12.

[DFR18] Jintai Ding, Scott R. Fluhrer, and Saraswathy RV. Complete attack on RLWE key exchange with reused keys, without signal leakage. In Willy Susilo and Guomin Yang, editors, *ACISP 18*, volume 10946 of *LNCS*, pages 467–486. Springer, Heidelberg, July 2018. doi:10.1007/978-3-319-93638-3_27.

[DGK24] Nir Drucker, Shay Gueron, and Dusan Kostic. A lean bike kem design for ephemeral key agreement. In *5th NIST Post-Quantum Cryptography Standardization Conference*. National Institute of Standards and Technology, 2024. URL: https://csrc.nist.gov/Presentations/2024/a-lean-bike-kem-design.

[Flu16] Scott Fluhrer. Cryptanalysis of ring-LWE based key exchange with key share reuse. Cryptology ePrint Archive, Report 2016/085, 2016. https://eprint.iacr.org/2016/085.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 537–554. Springer, Heidelberg, August 1999. doi:10.1007/3-540-48405-1_34.

[GJN20] Qian Guo, Thomas Johansson, and Alexander Nilsson. A key-recovery timing attack on post-quantum primitives using the Fujisaki-Okamoto transformation and its application on FrodoKEM. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 359–386. Springer, Heidelberg, August 2020. doi:10.1007/978-3-030-56880-1_13.

[GM23] Qian Guo and Erik Mårtensson. Do not bound to a single position: Near-optimal multi-positional mismatch attacks against kyber and saber. In Thomas Johansson and Daniel Smith-Tone, editors, *Post-Quantum Cryptography*, pages 291–320, Cham, 2023. Springer Nature Switzerland. doi:10.1007/978-3-031-40003-2_11.

[GMR20] Aurélien Greuet, Simon Montoya, and Guénaël Renault. Attack on LAC key exchange in misuse situation. In Stephan Krenn, Haya Shulman, and Serge Vaudenay, editors, *CANS 20*, volume 12579 of *LNCS*, pages 549–569. Springer, Heidelberg, December 2020. doi:10.1007/978-3-030-65411-5_27.

[HDV22] Loïs Huguenin-Dumittan and Serge Vaudenay. On ind-qcca security in the rom and its applications: Cpa security is sufficient for tls 1.3. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2022)*, pages 613–642. Springer, 2022. doi:10.1007/978-3-031-07082-2_22.

[HPP21] Julius Hermelink, Peter Pessl, and Thomas Pöppelmann. Fault-enabled chosen-ciphertext attacks on kyber. In Avishek Adhikari, Ralf Küsters, and Bart Preneel, editors, *Progress in Cryptology – INDOCRYPT 2021*, pages 311–334, Cham, 2021. Springer International Publishing. doi:10.1007/978-3-030-92518-5_15.

[HV20] Loïs Huguenin-Dumittan and Serge Vaudenay. Classical misuse attacks on NIST round 2 PQC - the power of rank-based schemes. In Mauro Conti, Jianying Zhou, Emiliano Casalicchio, and Angelo Spognardi, editors, *ACNS 20, Part I*, volume 12146 of *LNCS*, pages 208–227. Springer, Heidelberg, October 2020. doi:10.1007/978-3-030-57808-4_11.

[JMZ23] Haodong Jiang, Zhi Ma, and Zhenfeng Zhang. Post-quantum security of key encapsulation mechanism against cca attacks with a single decapsulation query. In *International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT 2023)*, pages 434–468. Springer, 2023. doi:10.1007/978-981-99-8730-6_14.

[MJZ22] Ruiqi Mi, Haodong Jiang, and Zhenfeng Zhang. Lattice reduction meets key-mismatch: New misuse attack on lattice-based nist candidate kems. Cryptology ePrint Archive, Paper 2022/1064, 2022. URL: https://eprint.iacr.org/2022/1064.

[MKB$^+$24] Puja Mondal, Suparna Kundu, Sarani Bhattacharya, Angshuman Karmakar, and Ingrid Verbauwhede. A practical key-recovery attack on lwe-based key-encapsulation mechanism schemes using rowhammer. In Christina Pöpper and Lejla Batina, editors, *Applied Cryptography and Network Security*, pages 271–300, Cham, 2024. Springer Nature Switzerland. doi:10.1007/978-3-031-54776-8_11.

[MN23] Alexander May and Julian Nowakowski. Too many hints – when LLL breaks LWE. In *Advances in Cryptology – ASIACRYPT 2023: 29th International Conference on the Theory and Application of Cryptology and Information Security, Guangzhou, China, December 4–8, 2023, Proceedings, Part IV*, page 106–137, Berlin, Heidelberg, 2023. Springer-Verlag. doi:10.1007/978-981-99-8730-6_4.

[Nat23] National Institute of Standards and Technology. Module-lattice-based key-encapsulation mechanism standard. Technical report, Department of Commerce, Washington, D.C., 2023. Federal Information Processing Standards Publication (FIPS) NIST FIPS 203 ipd. https://doi.org/10.6028/NIST.FIPS.203.ipd.

[OWT20] Satoshi Okada, Yuntao Wang, and Tsuyoshi Takagi. Improving key mismatch attack on NewHope with fewer queries. In Joseph K. Liu and Hui Cui, editors, *ACISP 20*, volume 12248 of *LNCS*, pages 505–524. Springer, Heidelberg, November / December 2020. doi:10.1007/978-3-030-55304-3_26.

[QCD19] Yue Qin, Chi Cheng, and Jintai Ding. A complete and optimized key mismatch attack on NIST candidate NewHope. In Kazue Sako, Steve Schneider, and Peter Y. A. Ryan, editors, *ESORICS 2019, Part II*, volume 11736 of *LNCS*, pages 504–520. Springer, Heidelberg, September 2019. doi:10.1007/978-3-030-29962-0_24.

[QCZ$^+$21] Yue Qin, Chi Cheng, Xiaohan Zhang, Yanbin Pan, Lei Hu, and Jintai Ding. A systematic approach and analysis of key mismatch attacks on lattice-based NIST candidate kems. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part IV*, volume 13093 of *Lecture Notes in Computer Science*, pages 92–121. Springer, 2021. URL: https://doi.org/10.1007/978-3-030-92068-5_4, doi:10.1007/978-3-030-92068-5_4.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005. doi:10.1145/1060590.1060603.

[RRCB20] Prasanna Ravi, Sujoy Sinha Roy, Anupam Chattopadhyay, and Shivam Bhasin. Generic side-channel attacks on CCA-secure lattice-based PKE and KEMs. *IACR TCHES*, 2020(3):307–335, 2020. https://tches.iacr.org/index.php/TCHES/article/view/8592. doi:10.13154/tches.v2020.i3.307-335.

[RRD$^+$23] Gokulnath Rajendran, Prasanna Ravi, Jan-Pieter D'Anvers, Shivam Bhasin, and Anupam Chattopadhyay. Pushing the limits of generic side-channel attacks on lwe-based kems - parallel pc oracle attacks on kyber kem and beyond. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2023(2):418–446, Mar. 2023. doi:10.46586/tches.v2023.i2.418-446.

[SAB$^+$20] Peter Schwabe, Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, and Damien Stehlé. CRYSTALS-KYBER. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions.

[SLZ24] Mingyao Shao, Yuejun Liu, and Yongbin Zhou. Pairwise and parallel: Enhancing the key mismatch attacks on kyber and beyond. ACM ASIA CCS 2024, 2024. URL: https://eprint.iacr.org/2023/887.

[TUX$^+$23] Yutaro Tanaka, Rei Ueno, Keita Xagawa, Akira Ito, Junko Takahashi, and Naofumi Homma. Multiple-valued plaintext-checking side-channel attacks on post-quantum kems. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2023(3):473–503, Jun. 2023. doi:10.46586/tches.v2023.i3.473-503.

[UXT$^+$22] Rei Ueno, Keita Xagawa, Yutaro Tanaka, Akira Ito, Junko Takahashi, and Naofumi Homma. Curse of re-encryption: A generic power/EM analysis on post-quantum KEMs. *IACR TCHES*, 2022(1):296–322, 2022. doi:10.46586/tches.v2022.i1.296-322.

[XIU$^+$21] Keita Xagawa, Akira Ito, Rei Ueno, Junko Takahashi, and Naofumi Homma. Fault-injection attacks against NIST's post-quantum cryptography round 3 KEM candidates. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part II*, volume 13091 of *LNCS*, pages 33–61. Springer, Heidelberg, December 2021. doi:10.1007/978-3-030-92075-3_2.

[ZJZ24] Biming Zhou, Haodong Jiang, and Yunlei Zhao. Cpa-secure kems are also sufficient for post-quantum tls 1.3. In *30th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT 2024)*, 2024. URL: https://eprint.iacr.org/2024/1360.