Improving Correlation Power Analysis on Masked CRYSTALS-Kyber with Lattice Attack

Yen-Ting Kuo[1] and Atsushi Takayasu[1]

The University of Tokyo, Tokyo, Japan
{kuruwakuo,takayasu-a}@g.ecc.u-tokyo.ac.jp

Abstract. Tosun and Savas (IEEE TIFS'23) proposed a non-profiling power analysis attack on *masked* ML-KEM, or CRYSTALS-Kyber. Their attack can recover a full secret key of Kyber with 7,000 power traces. Later, Tosun et al. (IEEE Access'24) claimed an improvement over the previous attack with only 550 traces, but the result is not convincing. In particular, their attack does not seem to recover a full secret key of masked Kyber; instead, it recovers only the absolute values for every coefficient of a secret key. Unfortunately, Tosun et al. did not provide convincing and efficient ways to recover the signs of every secret coefficient. In this paper, we show that 400 traces are sufficient to recover a full secret key of masked Kyber. This improvement is arguably significant, as the number of traces is only about 5% of a previous full key recovery attack by Tosun and Savas. The key technique for improvement is the use of a lattice embedding method. So far, there have been several known attacks that use Kannan's embedding method to reduce the number of traces for recovering a full secret key of Kyber. Specifically, these attacks recover only a partial secret key through power analysis attack and recover the remaining part by applying the embedding method. In contrast, we use not only recovered partial secret key but also recovered absolute values to recover the remaining part. For this purpose, we utilize an unusual embedding method that is a combination of Kannan's embedding and Bai-Galbraith's embedding. Our technique can also be applied to other post-quantum cryptosystems that use NTT-based multiplication. We demonstrate the applicability of our method to the first-order masking implementations of NTT-based variants of SABER and Dilithium, achieving full key recovery with 150 and 1,000 traces, respectively.

Keywords: CRYSTALS-Kyber · Lattice · Side-channel attack · Embedding technique

1 Introduction

1.1 Background

Kyber and (Module-)LWE. The advancement of quantum computing presents a serious threat to classical cryptographic protocols, such as RSA [RSA78] and elliptic curve cryptography [Mil86], as they can be efficiently broken using Shor's

algorithm [Sho94]. To address this looming quantum threat, the National Institute of Standards and Technology (NIST) has initiated the Post-Quantum Cryptography (PQC) standardization process. In the Key Encapsulation Mechanism (KEM) category, ML-KEM [NIS24], which is derived from CRYSTALS-Kyber [BDK+18], has been chosen as the primary algorithm by NIST. The security of Kyber relies on the Module Learning with Errors (Module-LWE) problem, which is a variant of the LWE problem. Given $(\mathbf{A}, \mathbf{t} = \mathbf{As} + \mathbf{e}) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$, where $\mathbf{A}$ is drawn from uniform random distribution and $\mathbf{e} \in \mathbb{Z}^m$ is a short error vector, the task of the LWE problem is to find a secret vector $\mathbf{s} \in \mathbb{Z}_q^n$. The standard approach for solving the LWE problem is called a primal attack, such as Kannan's embedding [Kan87] and Bai-Galbraith's embedding [BG14b]. Briefly speaking, Kannan's embedding (resp. Bai-Galbraith's embedding) is faster than the other embedding method if $\mathbf{e}$ is relatively larger (resp. shorter) than $\mathbf{s}$. Let $\mathcal{R} = \mathbb{Z}[x]/(x^n+1)$ and $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n+1)$. Given $(\mathbf{A}, \mathbf{t} = \mathbf{As}+\mathbf{e}) \in \mathcal{R}_q^{\ell \times k} \times \mathcal{R}_q^{\ell}$, where $\mathbf{e} \in \mathcal{R}^{\ell}$ is a small error polynomial, the task of the Module-LWE problem is to find a secret polynomial $\mathbf{s} \in \mathcal{R}_q^k$. In the case of Kyber, a secret polynomial $\mathbf{s}$ is small.

To accelerate the multiplication of polynomials, Module-LWE-based cryptosystems commonly employ the Number Theoretic Transform (NTT), serving as a specialized variant of the Fast Fourier Transform (FFT) adapted to finite fields. In an NTT-based polynomial multiplication, the coefficients of two polynomial are transformed to the NTT domain, where efficient pairwise multiplication is possible. In short, NTT and NTT^{-1} is a linear transformation; thus, there is a matrix $\mathbf{M}$ such that $\mathbf{s} = \mathbf{M}\hat{\mathbf{s}}$. Briefly speaking, given $\mathbf{A} \in \mathcal{R}_q^{\ell \times k}$, $\mathbf{s} \in \mathcal{R}_q^k$, and $\mathbf{e} \in \mathcal{R}^{\ell}$ in the normal domain, we first compute $\hat{\mathbf{A}} = \mathrm{NTT}(\mathbf{A})$, $\hat{\mathbf{s}} = \mathrm{NTT}(\mathbf{s})$, and $\hat{\mathbf{e}} = \mathrm{NTT}(\mathbf{e})$ in the NTT domain, where $\mathbf{s}$ is a secret key of Kyber. Then, we compute $\hat{\mathbf{t}} = \hat{\mathbf{A}} \circ \hat{\mathbf{s}} + \hat{\mathbf{e}}$ in the NTT domain and obtain $\mathbf{t} = \mathbf{As}+\mathbf{e} = \mathrm{NTT}^{-1}(\hat{\mathbf{t}})$ in the normal domain, where the overall computation is more efficient than the direct computation of $\mathbf{As}+\mathbf{e}$. Due to the efficiency and strong security guarantees of Kyber, it is widely regarded as a robust choice for securing communication in the post-quantum era.

CPA on Kyber. While Kyber's cryptographic foundations are secure against quantum adversaries, implementations of Kyber remain vulnerable to side-channel attacks, particularly power analysis attacks [PPM17, PP19, MWK+24, KT23, TMS24]. Side-channel attacks exploit the physical behavior of a device, including its power consumption patterns, to extract secret information. In the past, correlation power analysis (CPA) [KJJ99] and other side-channel techniques have demonstrated the potential to compromise the security of cryptographic implementations, even when protected by countermeasures. The focus on implementation security, especially against side-channel threats, has thus become a critical area of research as PQC algorithms transition from theoretical constructs to practical deployments.

A series of simple power analysis (SPA) [PPM17, PP19, HHP+21] targeting the NTT of Kyber were proposed that lead to full static key recovery. However, these attacks are still limited in that they either require extensive profiling

efforts or they are only applicable in specific scenarios like the encryption of ephemeral keys. On the other hand, Mujdei et al. [MWK+24] developed a CPA-based approach that successfully exploited leakage to recover a full secret key from incomplete NTT operations with 200 power traces. Afterwards, several studies [KT23, TS24, WXT24] have improved the attacks. Among them, Kuo and Takayasu [KT23] took an interesting approach, although the improvement is not very substantial. Kuo and Takayasu's attack consists of two steps. At first, they recover not a full secret key $\hat{\mathbf{s}} \in \mathcal{R}_q$ but only a partial secret key which we denote by a vector $\hat{\mathbf{s}}_{I_0} \in \mathbb{Z}_q^{\kappa}$ using CPA. In the second step, they recovered the remaining elements $\hat{\mathbf{s}}_{I_1} \in \mathbb{Z}_q^{n-\kappa}$ by solving the standard LWE problem. To this end, they utilized a simple fact, i.e., the relation $\mathbf{s} = \mathbf{M}\hat{\mathbf{s}}$ suggest $\mathbf{s} = \mathbf{M}_{I_0}\hat{\mathbf{s}}_{I_0} + \mathbf{M}_{I_1}\hat{\mathbf{s}}_{I_1}$, where all columns of $\mathbf{M}_{I_0} \in \mathbb{Z}_q^{n\times\kappa}$ and $\mathbf{M}_{I_1} \in \mathbb{Z}_q^{n\times(n-\kappa)}$ correspond to those of $\mathbf{M} \in \mathbb{Z}_q^{n\times n}$. Since we know $(\mathbf{M}_{I_1}, \mathbf{M}_{I_0}\hat{\mathbf{s}}_{I_0} = \mathbf{M}_{I_1}\hat{\mathbf{s}}_{I_1} - \mathbf{s}) \in \mathbb{Z}_q^{n\times(n-\kappa)} \times \mathbb{Z}_q^n$ due to the step 1 and $\mathbf{s} \in \mathbb{Z}^n$ is a short vector, we can use Kannan's embedding [Kan87] and recover $\hat{\mathbf{s}}_{I_1}$ if κ is sufficiently large. In other words, Kuo and Takayasu's approach suggests that the CPA step does not have to recover a full secret key. The state-of-the-art attack proposed by Wang et al. [WXT24] follows the two-step approach and requires only 15 traces.

CPA on Masked Kyber. Masking is an effective method to protect Kyber from power analysis attacks, and there are also examples applied on Kyber [BGR+21, OY23]. CPA on a masked version of cryptosystems typically requires much more effort to compromise. Although Tosun and Savas [TS24] proposed the first CPA attack on masked Kyber, it requires a minimum of 7,000 traces and estimated 48.5 days of analysis to recover a full key from the first-order protected Kyber using the standard second-order CPA.

Tosun et al. [TMS24] claimed that they found a method to recover a full secret key of masked implementations of Kyber with both the Montgomery and Plantard reduction methods. Their attack requires 850 and 550 traces to recover a full secret key, respectively. Moreover, they further claimed that they could use Kuo and Takayasu's two-step method and reduce the number of traces to 250. However, the result is not convincing in the sense that whether their attack can recover a full secret key is unclear. Tosun et al. used the absolute value prediction function in their CPA. Therefore, a hypothesis and its additive inverse get the same correlation score. Indeed, their experiment[1] consider the attack successful if not a full secret key $\hat{\mathbf{s}} = (\hat{s}_0, \ldots, \hat{s}_{n-1})$ but all absolute values $(|\hat{s}_0|, \ldots, |\hat{s}_{n-1}|)$ are recovered. Although we can recover the signs easily if the implementation is not masked [CKA+21], we do not know how to recover them efficiently if the implementation is masked.

Other non-Profiled Attacks. On the other hand, PC-oracle attacks target the Fujisaki-Okamoto (FO) transform [FO99]. Under normal circumstances, this feature prevents attacks by decrypting a message and re-encrypting it to ensure the result matches the original input. If they don't match, the system rejects the

[1] https://github.com/toluntosun21/ExploitingCentralReduction/blob/master/cw/attack/kyber_hocpa.ipynb

message. However, attackers can exploit unintended information leaks during this double-check process. By monitoring leaks related to the decrypted message, attackers can bypass these protections and launch key mismatch attacks [RRCB20] on the core encryption method used by the system.

Fluhrer [Flu16] originally established the framework for these attacks against lattice-based schemes. This was subsequently adapted to ML-KEM by [QCD19], establishing a baseline of 1855 queries, which corresponds to the number of traces required in a CPA. Optimization efforts continued with [QCZ+21] and [GM23], which reduced the requirement to 1776 and 1588 queries, respectively.

To protect the FO-Transform, several proposals for masked comparisons exist. Early attempts such as [OSPG18] and [BPO+20], were claimed insecure by [BDH+21]. Subsequently, [DHP+22] demonstrated that even the improved method in [BDH+21] was vulnerable to a practical higher-order horizontal collision attack. While this didn't technically violate security definitions, [DHP+22] responded by proposing a more robust higher-order solution. Recently, [DVV23] succeeded in optimizing the performance of these previous methods, but [HNPS24] show that 7000 chosen-ciphertexts and corresponding traces for realistic noise levels can compromise the security of [DVV23].

Other Cryptosystems. Power analysis targeting other cryptosystems that can be accelerated by NTT-based multiplication has also been proposed. For example, SABER [DKRV18] is another KEM whose security is based on the Module Learning with Rounding problem. While SABER was originally designed to leverage Toom-Cook and Karatsuba multiplications, there have been implementations that outperform the original design using NTT-based multiplication [CHK+21]. Ngo *et al.* [NDGJ21] introduced a machine learning-based method to break original SABER protected by first-order masking. Later, Ngo *et al.* [NDJ23] demonstrated another deep neural network attack on SABER with both masking and shuffling, which offers higher security. However, no known CPA attack has been proposed on NTT-based SABER protected by masking.

Dilithium [DKL+18], who is also based on the module-LWE problem, is another example that utilizes NTT-based polynomial multiplication. Chen *et al.* [CKA+21] developed a conservative scheme that reduces the size of the key guess space, claiming they can extract a secret key coefficient of unprotected Dilithium using 157 power traces. Later, Liu *et al.* [LLZ+24] achieved a 365-fold speed-up at the cost of increasing the number of power traces to 3,000. Similarly, no CPA attack has been proposed for Dilithium implementation with masking protection as far as we know.

1.2 Our Contribution

In this paper, we present improved power analysis attacks on masked CRYSTALS-Kyber, targeting its polynomial arithmetic operations, which are performed using the NTT. By leveraging new analysis techniques, we significantly reduce the number of traces required for a successful key recovery compared to previous approaches, which can be seen in Table 1. Our methods exploit the data-dependent

leakage inherent in the NTT-based polynomial multiplication and utilize advanced statistical techniques to efficiently correlate observed power traces with sensitive key material. We further demonstrate the applicability of our attacks across different Kyber parameter sets and discuss their implications for real-world implementations.

To achieve the low trace number requirement, we modify the two-step attack proposed by [KT23]. In the first step, the attack proceeds by recovering part of the NTT domain coefficients of the secret key. For the masked implementation of Kyber, it is quite tempting to use the same power model as Tosun *et al.* [TMS24] because of its simplicity and minimal trace requirements for recovering the absolute values. Another difference in the first step lies in the classification of the coefficients. In our attack, the coefficients will be split into three groups: confirmed ones, where we know the exact values; positive/negative ones, where we know only the absolute values; and unknown ones, where we have no information. Then, in the second step, we use a different embedding method to solve the LWE instance generated using these coefficients. As a result, we successfully decrease the number of traces required to perform a full-key recovery on masked Kyber to around 400 traces. Notice in Table 1, the actual number of traces required to attack masked Kyber using the method of Tosun *et al.* [TMS24] is 550, contrary to their claimed 250. This discrepancy arises because their best result combines a basic CPA attack with the lattice-based recovery technique described by Kuo and Takayasu [KT23]. However, that approach assumes the recovered coefficients have no sign ambiguity. In contrast, our experiment in Figure 2, which performs a negative correlation on masked Kyber, indicates that resolving this ambiguity requires significantly more than a few hundred traces.

Furthermore, our technique can also be applied to other cryptosystems that utilize NTT-based polynomial multiplication. We successfully migrated our attack to first-order masked SABER implemented by NTT-based multiplication and were able to recover the secret key using only 150 traces. As far as we know, this is the first CPA attack targeting NTT-based SABER protected by first-order masking. Another example is the Dilithium digital signature algorithm, which is also based on the Module-LWE problem like Kyber. Due to the large size of its hypothesis set, we adapted the Tunstall *et al.* attack [THM+07] and combined it with our lattice analysis. Overall, 1,000 traces were sufficient to recover the long-term secret key, which is only 20% of the traces required to recover all coefficients. Table 1 formalizes the above discussion and positions our study among the attacks existing in the literature.

A key advantage of our method lies in its ability to significantly reduce the lattice dimension, similar to the strategy used by Kuo and Takayasu [KT23] in their attack on unprotected Kyber. Unlike previous approaches that construct the LWE instance using only public data—resulting in larger, high-dimensional lattices—our technique exploits the shortness of the secret key in the normal domain and the structural leakage from Kyber's incomplete NTT. This enables us to construct a compact lattice while retaining all the necessary information for key recovery. Furthermore, our approach is not limited to first-order masking;

Table 1: Summary of state-of-the-art CPA attacks on Kyber, SABER and Dilithium. Simulation is performed by using ELMO for Kyber, and using Hamming weight model with $\sigma = 3$ for SABER and Dilithium.

Algorithm	Work	Target	Masked	Traces (un-/masked)
Kyber	This work	[KRSS19, BGR+21]	✓	✗/400 (Use simulation)
	[TMS24]	[KRSS19, BGR+21]	✓	✗/550 (Cannot recover sign)
	[TS24]	[KRSS19]	✓	150/7,000
	[WXT24]	[KRSS19]	✗	15/✗
	[MWK+24]	[KRSS19]	✗	200/✗
	[KT23]	[KRSS19]	✗	200/✗
SABER	This Work	[CHK+21, ACC+22]	✓	✗/150 (Use simulation)
Dilithium	This Work	[ABC+23]	✓	✗/1,000 (Use simulation)
	[LLZ+24]	[BDK+21]	✗	3,000/✗
	[CKA+21]	[BDK+21]	✗	157/✗

once enough coefficients are recovered, the same lattice reduction technique can be applied even under high-order masking protections.

Our another contribution is proposing a method to directly incorporate sign information into the lattice embedding itself. That is, compared to previous attacks, our method leverages additional information extracted from the absolute values of the coefficients to construct the LWE instance, thereby improving the ability to recover the secret key. This enhancement allows us to achieve full key recovery with only 400 traces, marking a significant improvement over previous full-key recovery attacks, which required approximately 7,000 traces.

We acknowledge that our evaluation uses simulated power traces rather than physical measurements from hardware. However, our key contribution lies in the lattice-based recovery strategy. The CPA phase closely follows established techniques from prior work, and our improvements stem primarily from innovations in the lattice construction. For this reason, we focused our evaluation on demonstrating the effectiveness of our lattice attack rather than optimizing trace acquisition from physical devices.

Organization. In Section 2, we introduce the CRYSTALS-Kyber cryptosystem and review relevant side-channel techniques and masking in lattice-based cryptography. In Section 3, we present our improved attack strategy, detailing its application to masked Kyber implementations. Section 4 evaluates the attack on different noise level, including experimental setups and performance metrics. Finally, we discusses implications for other schemes like SABER and Dilithium in Section A.

2 Preliminaries

We write vectors in lower-case bold, e.g. $\mathbf{b}$, and matrices in upper-case bold, e.g. $\mathbf{A}$. We write $\mathbf{I}_m$ for the $m \times m$ identity matrix over whichever base ring

is implied from context. We write $\mathbf{0}_{m\times n}$ for the $m \times n$ all zero matrix. If the dimensions are clear from the context, we may omit the subscripts.

2.1 Lattices

Let $\mathbf{B} = [\mathbf{b}_1^\top, ..., \mathbf{b}_n^\top]^\top \in \mathbb{Z}^{n\times m}$ be an integer matrix. We denote by

$$\Lambda(\mathbf{B}) \coloneqq \{\alpha_1\mathbf{b}_1 + ... + \alpha_n\mathbf{b}_n \mid \alpha_i \in \mathbb{Z}\}$$

the lattice generated by $\mathbf{B}$. If the rows of $\mathbf{B}$ are linearly independent, $\mathbf{B}$ is a *basis matrix* of $\Lambda(\mathbf{B})$. The same lattice $\Lambda(\mathbf{B})$ can be represented using different bases, and one can be obtained from another by multiplying it by unimodular matrix. The number of rows n in any basis matrix of some lattice Λ is called the *rank* of Λ. The *determinant* of a lattice Λ with basis matrix $\mathbf{B}$ is defined as $\det(\Lambda) \coloneqq \sqrt{\det(\mathbf{B}\mathbf{B}^\top)}$. The determinant does not depend on the choice of basis. We also denote by $\lambda_i(\Lambda)$ the *i-th successive minimum* of Λ. A lattice vector $\mathbf{v} \in \Lambda$ such that $\|\mathbf{v}\| = \lambda_1(\Lambda)$ is called the *shortest vector* of Λ. $\lambda_1(\Lambda)$ can be estimated by the Gaussian heuristic,

$$\lambda_1(\Lambda) \approx \sqrt{\frac{n}{2\pi e}}\det(\Lambda)^{1/n}.$$

Lattice Problems. Let $\gamma \geq 1$, given a basis $\mathbf{B}$ of Λ, the γ-shortest vector problem (SVP_γ) asks us to find $\mathbf{s} \in \Lambda(\mathbf{B})$ such that $\|\mathbf{s}\| \leq \gamma \cdot \lambda_1(\Lambda)$. Given a basis $\mathbf{B}$ of Λ with $\lambda_2(\Lambda(\mathbf{B})) > \gamma \cdot \lambda_1(\Lambda(\mathbf{B}))$ guaranteed, the γ-unique shortest vector problem (uSVP_γ) asks us to find the non-zero shortest vector $\mathbf{s} \in \Lambda$. Given a basis $\mathbf{B}$ of Λ and a target vector $\mathbf{t} \in \mathbb{R}^d$ such that the distance between $\mathbf{t}$ and $\Lambda(\mathbf{B})$ can be bounded by $\lambda_1(\Lambda(\mathbf{B}))/\gamma$, the γ-bounded distance decoding (BDD_γ) problem asks us to find vector $\mathbf{b} \in \Lambda(\mathbf{B})$ closest to $\mathbf{t}$. Lattice problems are harder to solve for smaller γ.

There exist algorithms, such as the LLL algorithm [LLL82], that can be utilized to solve SVPγ and uSVPγ, providing relatively short vectors within practical time. The BKZ algorithm, introduced by Schnorr and Euchner [CN11], trades off computing time with output quality using exact algorithms and the LLL algorithm as its subroutines.

2.2 LWE Problem

Learning with errors (LWE) problem [Reg05] and its extension over rings [LPR10] or modules are the basis of multiple NIST PQC candidates.

Let $n \in \mathbb{Z}$ denote the dimension, $m \in \mathbb{Z}$ denote the sample number and $q \in \mathbb{Z}$ denote the modulus. Let χ_s and χ_e denote two independent probability distributions on $\mathbb{Z}_q$ and $\mathbb{Z}$ with standard deviations $\sigma_s \in \mathbb{R}$ and $\sigma_e \in \mathbb{R}$, respectively. The search version of LWE problem is defined as follow:

Definition 1 (LWE problem). *Let* $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ *be a matrix whose entries are in* $\mathbb{Z}_q$, $\mathbf{s} \in \mathbb{Z}_q^n$ *be a secret vector whose entries are sampled from* χ_s, *and* $\mathbf{e} \in \mathbb{Z}_q^m$ *be an error vector whose entries are sampled from* χ_e. *The vector* $\mathbf{t} \in \mathbb{Z}_q^m$ *is calculated by* $\mathbf{t} \coloneqq \mathbf{A}\mathbf{s} + \mathbf{e} \mod q$. *The LWE problem is to the recover vector* $\mathbf{s}$ *from* $(\mathbf{A}, \mathbf{t}) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$.

The standard LWE problem proposed by Regev [Reg05] chooses secret vectors $\mathbf{s} \in \mathbb{Z}_q^n$ uniformly. In contrast, the binary LWE chooses the secret vector $\mathbf{s}$ uniformly from $\{0, 1\}$. We can solve the LWE problem by the embedding techniques who reduces LWE to a uSVP$_\gamma$ and then applying the BKZ algorithm. In this section, we recall three known embedding techniques.

Embedding techniques. Kannan [Kan87] proposed an embedding technique which reduces the LWE problem to the uSVP problem. We first transform the LWE problem into a BDD problem in which the target vector is $\mathbf{t}$ and the basis of $\Lambda(\mathbf{B}_{\mathsf{BDD}})$ is

$$\mathbf{B}_{\mathsf{BDD}} = \begin{bmatrix} \mathbf{I}_n & \mathbf{A}' \\ \mathbf{0} & q\mathbf{I}_{m-n} \end{bmatrix},$$

where $[\mathbf{I}_n \mid \mathbf{A}']$ denotes the reduced row echelon matrix of $\mathbf{A}^\top$, which can be easily calculated by Gaussian elimination. We can then solve the BDD of $\Lambda(\mathbf{B}_{\mathsf{BDD}})$ with respect to the target point $\mathbf{t}$ which reveals $\mathbf{s}$.

Alternatively, we can reduce this BDD to uSVP by considering a basis matrix $\mathbf{B}_{\mathsf{Kan}}$, which is a special case of *half-twisted embedding* defined later in this section. The equation $\mathbf{t} = \mathbf{A}\mathbf{s} + \mathbf{e} \mod q$ can be written as $\mathbf{t} = \mathbf{A}\mathbf{s} + \mathbf{e} + q\mathbf{r}$, where $\mathbf{r} \in \mathbb{Z}_q^n$, so there exists a row vector $[-\mathbf{s}^\top \mid -\mathbf{r}^\top \mid 1] \in \mathbb{Z}_q^{m+1}$ such that the shortest vector in $\Lambda(\mathbf{B}_{\mathsf{Kan}})$ is $[-\mathbf{s}^\top \mid -\mathbf{r}^\top \mid 1] \cdot \mathbf{B}_{\mathsf{Kan}} = [\mathbf{e}^\top \mid 1] \in \mathbb{Z}_q^{n+1}$.

Bai and Galbraith [BG14b] introduced an embedding better suited for binary-LWE or instances where the secret and error have different magnitudes. They incorporate a scaling factor $\nu = \sigma_e/\sigma_s$ to balance their norms. The embedded shortest vector becomes $[-\nu\mathbf{s}^\top \mid \mathbf{e}^\top \mid \lambda]$. This method increases the lattice volume to widen the gap between successive minima, which is favorable when $\sigma_s < \sigma_e$.

Wang et al. [WWTT18] proposed a flexible hybrid method that combines Kannan's and Bai-Galbraith's embeddings. Let a positive number $n_\top \in [0, n]$, they divided $\mathbf{A} = [\mathbf{A}_{\mathsf{BG}} \mid \mathbf{A}_{\mathsf{Kan}}]$, where into matrices $\mathbf{A}_{\mathsf{BG}} \in \mathbb{Z}^{m \times n_\top}$ and $\mathbf{A}_{\mathsf{Kan}} \in \mathbb{Z}^{m \times (n - n_\top)}$, and divided $\mathbf{s} = [\mathbf{s}_{\mathsf{BG}} \mid \mathbf{s}_{\mathsf{Kan}}]$ into vectors $\mathbf{s}_{\mathsf{BG}} \in \mathbb{Z}_q^{n_\top}$ and $\mathbf{s}_{\mathsf{Kan}} \in \mathbb{Z}_q^{n - n_\top}$.

By applying Kannan's embedding to $\mathbf{A}_{\mathsf{Kan}}$ and Bai-Galbraith's embedding to $\mathbf{A}_{\mathsf{BG}}$, they reduced an LWE problem to a uSVP$_\gamma$ in a lattice $\Lambda(\mathbf{B}_{\mathsf{HT}})$ with a basis matrix

$$\mathbf{B}_{\mathsf{HT}} = \left[\begin{array}{c|cc|c} \nu\mathbf{I}_{n_\top} & \multicolumn{2}{c|}{\mathbf{A}_{\mathsf{BG}}^\top} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{I}_{n-n_\top} & \mathbf{A}'_{\mathsf{Kan}} & \mathbf{0} \\ & \mathbf{0} & q\mathbf{I}_{m-(n-n_\top)} & \\ \hline \mathbf{0} & \multicolumn{2}{c|}{\mathbf{t}^\top} & \lambda \end{array}\right],$$

where λ is the embedding parameter and $\mathbf{A}'_{\mathrm{Kan}}$ is in row-echelon form. The embedding is equivalent to Kannan's embedding for $n_\top = 0$, and it is the same

as Bai-Galbraith's embedding for $n_\top = n$. The lattice $\Lambda(\mathbf{B}_{\mathsf{HT}})$ contains a vector $[-\nu\mathbf{s}_{\mathsf{BG}} \mid \mathbf{e} \mid \lambda]$.

2.3 CRYSTALS-Kyber

Table 2: Parameter sets for Kyber [SAB+20].

name	n	k	q	η_1	η_2
Kyber512	256	2	3329	3	2
Kyber768	256	3	3329	2	2
Kyber1024	256	4	3329	2	2

CRYSTALS-Kyber [NIS24] is a Key Encapsulation Mechanism (KEM) submitted to the NIST standardization process, and it is among the four confirmed candidates to be standardized. The security of Kyber is based on the module-LWE problem. For the three parameter sets in the proposal, Kyber512, Kyber768, and Kyber1024, the parameters are all set to $n = 256$ and $q = 3329$. For most parameters $\eta = 2$ is used, except for Kyber512, where $\eta = 3$. The parameter sets differ in their module dimension $k = 2, 3$, and 4 respectively. The three parameter sets listed in Table 2.

Let $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n+1)$ be the polynomial ring of polynomials modulo x^n+1. For any ring $\mathcal{R}$, $\mathcal{R}^{\ell\times k}$ denotes the ring of $\ell \times k$-matrices over $\mathcal{R}$. We also simplify $\mathcal{R}^{\ell\times 1}$ to $\mathcal{R}^{\ell}$ if there is no ambiguity.

Kyber consists of the CCA2-KEM Key Generation, PKE and CCA2-KEM-Encryption, and CCA2-KEM-Decryption algorithms, which are summarized in Algorithms 1, 2, 3 and 4, respectively.

Algorithm 1 Kyber-CCA2-KEM Key Generation (simplified)

Output: Public key pk, secret key sk

1: Choose uniform seeds ρ, σ, z
2: $\mathcal{R}_q^{k\times k} \ni \hat{\mathbf{A}} \leftarrow \mathtt{Sample}_{\mathcal{U}}(\rho)$
3: $\mathcal{R}^{k} \ni \mathbf{s}, \mathbf{e} \leftarrow \mathtt{Sample}_{\beta_\eta}(\sigma)$
4: $\hat{\mathbf{s}} \leftarrow \mathrm{NTT}(\mathbf{s})$
5: $\hat{\mathbf{t}} \leftarrow \hat{\mathbf{A}} \circ \hat{\mathbf{s}} + \mathrm{NTT}(\mathbf{e})$
6: **return** $(pk := (\hat{\mathbf{t}}, \rho), sk := (\hat{\mathbf{s}}, pk, \mathrm{Hash}(pk), z))$

Here, $\mathtt{Sample}_{\mathcal{U}}$ is sampling the coefficients from an uniform distribution and $\mathtt{Sample}_{\beta_\eta}$ is sampling the coefficients from a binomial distribution ranged from $-\eta$ to η. Hash is a hash function of 32-byte output. `Encode` will encode a byte array into an array of 1-bit integers and then multiply it by $q/2$. The key derivation function is denoted by `KDF`.

Algorithm 2 Kyber-PKE Encryption `PKE.Enc` (simplified)

Input: Public key $pk = (\hat{\mathbf{t}}, \rho)$, message m, seed τ
Output: Ciphertext c

$\mathcal{R}_q^{k \times k} \ni \hat{\mathbf{A}} \leftarrow \mathtt{Sample}_{\upsilon}(\rho)$
$\mathcal{R}^k \ni \mathbf{r}, \mathbf{e}_1, \mathcal{R}_q \ni e_2 \leftarrow \mathtt{Sample}_{\beta_\eta}(\tau)$
$\mathbf{u} \leftarrow \mathrm{NTT}^{-1}(\hat{\mathbf{A}}^\top \circ \mathrm{NTT}(\mathbf{r})) + \mathbf{e}_1$
$v \leftarrow \mathrm{NTT}^{-1}(\hat{\mathbf{t}}^\top \circ \mathrm{NTT}(\mathbf{r})) + e_2 + \mathtt{Encode}(m)$
return $c \coloneqq (\mathbf{u}, v)$

In these algorithms, and in the rest of this paper, the notation $a \circ b$ means "pairwise" multiplication of polynomials, or vectors of polynomials, in the NTT domain. For example, if $a = (a_0, a_1)$ and $b = (b_0, b_1)$, $a \circ b = (a_0 b_0, a_1 b_1)$.

Kyber uses a variant of the Fujisaki-Okamoto transform [FO99] to build an IND-CCA2 secure KEM scheme. This transform applies an additional re-encryption of the decrypted message, using the same randomness as used for the encryption of the received ciphertext. The decryption is only valid if the re-computed ciphertext matches the received ciphertext.

Algorithm 3 Kyber-CCA2-KEM Encapsulation (simplified)

Input: Public key $pk = (\hat{\mathbf{t}}, \rho)$
Output: Ciphertext c, shared key K

Choose uniform m
$(\bar{K}, \tau) \leftarrow \mathtt{Hash}(m \parallel \mathtt{Hash}(pk))$
$c \leftarrow \mathtt{PKE.Enc}(pk, m, \tau)$
$K \leftarrow \mathtt{KDF}(\bar{K} \parallel \mathtt{Hash}(c))$
return (c, K)

Algorithm 4 Kyber-CCA2-KEM Decapsulation (simplified)

Input: Secret key $sk = (\hat{\mathbf{s}}, pk, h, z)$, ciphertext $c = (\mathbf{u}, v)$
Output: Shared key K

$m \leftarrow \mathtt{Decode}(v - \mathtt{NTT}^{-1}(\hat{\mathbf{s}}^\top \circ \mathtt{NTT}(\mathbf{u})))$
$(K, \tau) \leftarrow \mathtt{Hash}(m \parallel h)$
$c' \leftarrow \mathtt{PKE.Enc}(pk, m, \tau)$
if $c = c'$ **then**
    return $K \coloneqq \mathtt{KDF}(K \parallel \mathtt{Hash}(c))$
else
    return $K \coloneqq \mathtt{KDF}(z \parallel \mathtt{Hash}(c))$
end if

2.4 Number Theoretic Transform

For lattice-based schemes using polynomial rings, polynomial multiplications in en-/decryption are the most computationally expensive step. The Number Theoretic Transform (NTT) is a technique that can achieve efficient computation for those multiplications.

The NTT is similar to the Discrete Fourier Transform (DFT), but instead of over the field of complex numbers, it operates over a prime field $\mathbb{Z}_q$. It can be seen as a mapping between the coefficient representation of a polynomial from $\mathcal{R}_q$, called the normal domain, to the evaluation of the polynomial at the n-th roots of unity, called the NTT domain. This bijective mapping is typically referred to as forward transformation. The mapping from the NTT domain to the normal domain is referred to as backward transformation or inverse NTT. In the NTT domain, the multiplication of polynomials can be achieved by pointwise multiplication, which is much cheaper than multiplication in the normal domain. Typically, one would perform the forward transformation, multiply the polynomials pointwisely in the NTT domain, and go back using the backward transformation. For $\mathcal{R}_q$ with a $2n$-th primitive root of unity ζ, the NTT transformation of an n-degree polynomial $f = \sum_{i=0}^{n-1} f_i x^i$ is defined as:

$$\hat{f} = \mathrm{NTT}(f) = \sum_{i=0}^{n-1} \hat{f}_i x^i, \quad \text{where} \quad \hat{f}_i = \sum_{j=0}^{n-1} f_j \zeta^{(2i+1)\cdot j}.$$

Similarly,

$$f = \mathrm{NTT}^{-1}(\hat{f}) = \sum_{i=0}^{n-1} f_i x^i, \quad \text{where}$$
$$f_i = n^{-1} \sum_{j=0}^{n-1} \hat{f}_j \zeta^{-i\cdot(2j+1)}.$$

The NTT transform and its inverse can be applied efficiently by using a chaining of $\log_2 n$ butterflies. It is a divide and conquer technique that splits the input in half in each step and solves two problems of size $n/2$. We use the Cooley-Tukey butterfly [CT65] with decimation in time to implement the NTT transform, with the output being in bit-reversed order. Notice that both NTT and inverse NTT are a linear transform, thus they can be expressed by matrix multiplications, e.g.

$$[f_i]^\top = \mathbf{M}[\hat{f}_i]^\top \tag{1}$$

for some $n \times n$ matrix $\mathbf{M}$.

Incomplete NTT. Kyber uses an NTT-friendly ring. But in Kyber, only n-th primitive roots of unity exist, therefore the modulus polynomial $x^n + 1$ only factors into polynomials of degree 2. Hence, the last layer between nearest neighbors of the NTT is skipped and in NTT domain multiplication is not purely pointwise, but multiplications of polynomials of degree 1. That is, the Kyber ring is

effectively $\mathbb{F}_{q^2}[y]/(y^{128}+1)$, where $\mathbb{F}_{q^2}$ is the field $\mathbb{Z}_q[x]/(x^2-\zeta)$. As a result, the $\mathbf{M}$ in above equation 1 is actually an $n/2$ by $n/2$ matrix. Also note that in Kyber, polynomials in the NTT domain are always considered in bit-reversed order. Therefore, in the following bit-reversal is implicitly expected in the NTT domain and indices for NTT-coefficients are noted in regular order.

2.5 Correlation Power Analysis

The objective of correlation power analysis (CPA) [KJJ99] is to uncover secret keys from cryptographic devices by analyzing a large number of power traces recorded during the decryption of various plaintexts. The success rate of CPA is influenced by both the quality and quantity of the traces. CPA is the most widely used power analysis attack because it does not require extensive knowledge of the targeted devices. Additionally, it can successfully reveal the secret key even when the recorded power traces are highly noisy.

The first step in CPA involves selecting an intermediate result of the cryptographic algorithm executed by the target device. This intermediate value is represented as a function $g(d,k)$, where d is a known, non-constant data value, typically the plaintext or ciphertext, and k is a small portion of the secret key.

Next, the attacker measures the power consumption of the device while it processes different data blocks during encryption or decryption. The data values associated with each block are known to the attacker and are denoted by the vector $\mathbf{d} = (d_1, ..., d_D)^\top$, where D is the number of data blocks.

The power consumption during each encryption or decryption run is recorded as a power trace, denoted by the vector $\mathbf{t}_i^\top = (t_{i,1}, ..., t_{i,T})$, where T is the length of the trace (the number of time points recorded). All the traces are organized into a matrix $\mathbf{T}$ of size $D \times T$, where each row corresponds to a trace for a particular data block. The correct alignment of these traces is crucial, ensuring that each column t_j of $\mathbf{T}$ corresponds to the same operation across all data blocks.

The next step involves calculating hypothetical intermediate values for all possible key hypotheses. The attacker considers multiple possible values for the key, denoted by the vector $\mathbf{k} = (k_1, ..., k_K)$, where K is the total number of key hypotheses. For each key hypothesis and each data block, the attacker computes a hypothetical intermediate value, resulting in a matrix $\mathbf{V}$ of size $D \times K$. The hypothetical intermediate values $\mathbf{V}$ is then mapped to hypothetical power consumption values using prediction function like the Hamming-weight or Hamming-distance models. This mapping results in a matrix $\mathbf{H}$ of the same size as $\mathbf{V}$.

Finally, we compare the hypothetical power consumption values in $\mathbf{H}$ with the actual power traces in $\mathbf{T}$. The result of this comparison is a matrix $\mathbf{R}$ of size $K \times T$, where each element $r_{i,j}$ contains the result of the comparison between the i-th column of $\mathbf{H}$ and j-th column of $\mathbf{T}$. The comparison is done based on

the Pearson correlation coefficient (PCC),

$$r_{i,j} = \frac{\sum_{d=1}^{D}(h_{d,i} - \bar{h}_i) \cdot (t_{d,j} - \bar{t}_j)}{\sqrt{\sum_{d=1}^{D}(h_{d,i} - \bar{h}_i)^2 \cdot \sum_{d=1}^{D}(t_{d,j} - \bar{t}_j)^2}}.$$

The goal is to identify the correct key hypothesis by finding the maximum value in $\mathbf{R}$. The highest value indicates the best match between the hypothetical and actual power consumption, revealing both the correct key and the specific time during execution when the intermediate value was processed.

Second-Order CPA In Second-Order CPA, the attacker identifies two distinct leakage points within a single power trace. These points typically correspond to the manipulation of two different shares of a masked intermediate value. The objective is to derive a combined leakage trace that is directly correlated with the unmasked sensitive data. The choice of the appropriate leakage model is crucial, as it dictates how hypothetical power consumption values are generated for correlation.

For attacks utilizing Pearson's correlation coefficient, the *mean-free product* is widely regarded as the most effective method for combining leakage samples from different shares [PRB09, SVO+10]. This combination function accounts for the mean power consumption, thereby removing constant offsets and enhancing the signal-to-noise ratio. The formula for the mean-free product of two samples t_{i,t_1} and t_{i,t_2} from a single power trace is given by:

$$\mathcal{C}(t_1, t_2) = (t_{i,t_1} - \mathrm{E}[t_{i,t_1}]) \cdot (t_{i,t_2} - \mathrm{E}[t_{i,t_2}])$$

where E denotes the mean of the power consumption at a specific time point across the entire set of collected traces.

CPA on Kyber. Here we recall two of the previous attacks that are related to our work. Kuo and Takayasu [KT23] presented a two-step attack on CRYSTALS-Kyber. In the first step, they utilize CPA to recover a *partial* secret key from the power consumption during polynomial multiplications in the decryption process. Instead of recovering the full secret key $\hat{\mathbf{s}}$, they recover a vector denoted as $\hat{\mathbf{s}}_{I_0}$, which represents a subset of the NTT domain coefficients. The second step involves recovering the *remaining* unknown elements of the secret key, denoted as $\hat{\mathbf{s}}_{I_1}$, by solving an LWE problem. This is achieved by exploiting the linear relationship between the normal domain secret key $\mathbf{s}$ and its NTT domain representation $\hat{\mathbf{s}}$, expressed as $\mathbf{s} = \mathbf{M}\hat{\mathbf{s}}$, where M is the inverse NTT matrix. This equation is then rearranged into an LWE instance. Kannan's embedding technique is then applied to solve this LWE problem and recover the remaining coefficients. The authors concluded that at least 39 (for Kyber512) or 38 (for Kyber768/1024) recovered coefficients from the CPA step are needed for full key recovery using a BKZ algorithm with a block size of 50.

Tosun *et al.* [TMS24] proposed a non-profiling power analysis attack on masked CRYSTALS-Kyber, primarily targeting its use of central reduction.

Their approach introduced a novel absolute value prediction function to exploit a vulnerability in the central reduction algorithm. This created a strong dependency between power consumption and the absolute value of sensitive intermediate values. However, a key limitation of their method is that the absolute value function meant that a secret coefficient and its additive inverse would yield the same correlation score. Consequently, their attack could recover only the absolute values for every coefficient of the secret key, not their signs. While they claimed an improvement requiring 550 traces for full key recovery, this result was not convincing as it didn't efficiently address sign recovery in masked implementations. Their best result, claiming 250 traces, combined a basic CPA with a lattice-based technique, but assumed no sign ambiguity, which is problematic for masked Kyber. We will also detail these two attacks in the following sections.

3 Lattice Attack

In this section, we describe how to embed the sign information into the lattice analysis of Kuo and Takayasu's attack [KT23] by using the additional information that some of the coefficients in the NTT form they got could be either positive or negative.

3.1 The Kuo-Takayasu's Attack

Kuo and Takayasu [KT23] proposed a two-step attack on Kyber. First, they recovered some portions of secret keys using correlation power analysis. Next, they showed that the remaining secrets can be recovered by solving the learning with errors (LWE) problem. Precisely, let $\mathbf{M} \in \mathbb{Z}_q^{128\times 128}$ be the inverse NTT matrix as we mentioned in Equation (1). Suppose we have recovered $128 - \xi$ coefficients in $\hat{\mathbf{s}}_i$, one of the groups of the NTT domain secret key $\hat{\mathbf{s}}$, from the polynomial multiplication $\hat{\mathbf{s}} \circ \hat{\mathbf{u}}$, i.e., we need to recover the remaining ξ coefficients. Let $I_0 = \{i_{0,0}, i_{0,1}, ..., i_{0,127-\xi}\}$ be the indices that are successfully recovered in the CPA step, and $I_1 = \{i_{1,0}, i_{1,1}, ..., i_{1,\xi-1}\}$ be the indices that are still unknown, then the inverse NTT, $\mathrm{NTT}^{-1}(\hat{\mathbf{s}}_i) = \mathbf{M}\hat{\mathbf{s}} = \mathbf{s} \mod q$ can be split into two halves as followed:

$$\mathbf{M}_{I_0}\hat{\mathbf{s}}_{I_0} + \mathbf{M}_{I_1}\hat{\mathbf{s}}_{I_1} = \mathbf{s} \mod q,$$

where $\mathbf{M}_{I_0} := [\mathbf{m}_{i_{0,0}} \mid \mathbf{m}_{i_{0,1}} \mid ... \mid \mathbf{m}_{i_{0,127-\xi}}] \in \mathbb{Z}_q^{128\times(128-\xi)}$ is a matrix whose columns are those of $\mathbf{M}$ with indices in I_0, and $\mathbf{m}_i$ denotes the i-th column of $\mathbf{M}$. Similarly, $\hat{\mathbf{s}}_{I_0} = [\hat{s}_{i_{0,0}}, ..., \hat{s}_{i_{0,127-\xi}}]$ is a vector whose entries are those of $\hat{\mathbf{s}}$ with indices in I_0. The definition for $\mathbf{M}_{I_1}$ and $\hat{\mathbf{s}}_{I_1}$ are analogous. From the CPA, they recover a vector $\tilde{\mathbf{s}}_{I_0} = \hat{\mathbf{s}}_{I_0}$ and want to recover the unknown $\hat{\mathbf{s}}_{I_1}$.

Notice that $\mathbf{s}$ is a short vector since it is the secret key sampled from β_{η_1}. By calling the known vector $\mathbf{t} = \mathbf{M}_{I_0}\tilde{\mathbf{s}}_{I_0}$, the known basis $\mathbf{A} = -\mathbf{M}_{I_1}$, and an unknown vector $\mathbf{s}_{I_1} = \hat{\mathbf{s}}_{I_1}$, we now have $\mathbf{t} = \mathbf{A}\mathbf{s}_{I_1} + \mathbf{s} \mod q$, which is exactly the definition of an LWE problem. Compared to the original module-LWE problem

in Kyber, this problem becomes simpler since the rank of $\mathbf{A}$ is less than the original one.

Then, they used the standard technique of Kannan's embedding introduced in Section 2.2 to solve the LWE problem and concluded that we need at least 39 or 38 recovered coefficients for Kyber512 or Kyber768/1024 so that we can have a fully recovered secret key when using the BKZ algorithm of block size 50 to solve the reduced uSVP_γ problem.

3.2 Proposed Lattice Attack

In the CPA attack proposed by Kuo and Takayasu, they mentioned that if the correct coefficients $(\hat{s}_{2i}, \hat{s}_{2i+1})$ yield a high score, then it is likely that its additive negative $(q-\hat{s}_{2i}, q-\hat{s}_{2i+1})$ will also yield a high score. It is difficult to determine which of these potentially positive or negative coefficient pairs is the correct one using a pure CPA method. To enhance the lattice analysis, we aim to incorporate these potentially positive or negative coefficients into the lattice.

Assuming that among the 128 coefficients in the NTT domain, there are n_{I_0} coefficients that are recovered from the CPA attack, and n_{I_1} coefficients that we only know its absolute value, which we call a positive/negative coefficient. We rearrange and split the vector $\hat{\mathbf{s}}$ into $[\hat{\mathbf{s}}_{I_0} \mid \hat{\mathbf{s}}_{I_1} \mid \hat{\mathbf{s}}_{I_2}]$, where $\hat{\mathbf{s}}_{I_0}$ is a length n_{I_0} vector whose entries are the coefficients that are recovered, $\hat{\mathbf{s}}_{I_1}$ is a length n_{I_1} vector whose entries are the coefficients that we only know the absolute value of, and $\hat{\mathbf{s}}_{I_2}$ is a length $128 - n_{I_0} - n_{I_1}$ vector whose entries are the coefficients that are still unknown.

Let $\mathbf{M}$ be the inverse NTT matrix of equation (1), we rearrange and split it by the similar way into $[\mathbf{M}_{I_0} \mid \mathbf{M}_{I_1} \mid \mathbf{M}_{I_2}]$. The inverse NTT transform of $\hat{\mathbf{s}}$, $\mathbf{M}\hat{\mathbf{s}} = \mathbf{s} \mod q$, can be written as

$$\mathbf{s} = \mathbf{M}_{I_0}\hat{\mathbf{s}}_{I_0} + \mathbf{M}_{I_1}\hat{\mathbf{s}}_{I_1} + \mathbf{M}_{I_2}\hat{\mathbf{s}}_{I_2} \mod q. \tag{2}$$

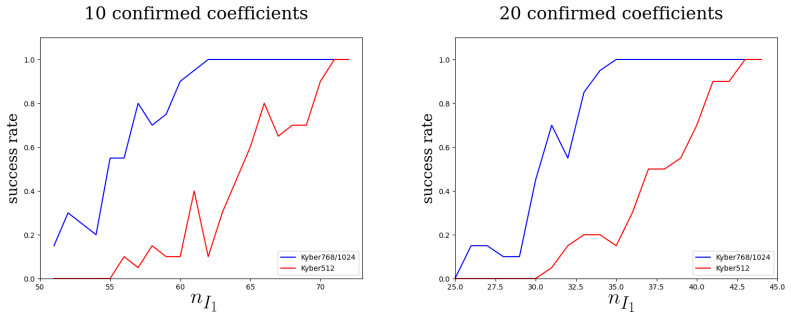
Suppose we have recovered a vector $\tilde{\mathbf{s}}_{I_0} = \hat{\mathbf{s}}_{I_0}$ from the CPA attack, and only recover a vector $\tilde{\mathbf{s}}_{I_1} = |\hat{\mathbf{s}}_{I_1}|$ for which we are uncertain about the sign. By letting $\mathbf{M}_{I_1} = \left[\mathbf{m}_0 \mid \mathbf{m}_1 \mid \cdots \mid \mathbf{m}_{n_{I_1}-1}\right]$, where $\mathbf{m}_i$ represents the i-th column of $\mathbf{M}_{I_1}$, and $\tilde{\mathbf{s}}_{I_1} = [\tilde{s}_0, \tilde{s}_1,$
$\ldots, \tilde{s}_{n_{I_1}-1}]$, we can express the vector $\mathbf{M}_{I_1}\hat{\mathbf{s}}_{I_1} = \widetilde{\mathbf{M}}_{I_1}\mathbf{d}$, where

$$\widetilde{\mathbf{M}}_{I_1} := \left[\tilde{s}_0\mathbf{m}_0\ \tilde{s}_1\mathbf{m}_1\ \ldots\ \tilde{s}_{n_{I_1}-1}\mathbf{m}_{n_{I_1}-1}\right],$$

and $\mathbf{d} = [d_0, d_1, ..., d_{n_{I_1}-1}]$ where $d_i \in \{1, -1\}$ corresponds to the coefficient being positive or negative. If we call the known vector $\mathbf{M}_{I_0}\hat{\mathbf{s}}_{I_0}$ by $\mathbf{t}$, equation (2) can be written as

$$\mathbf{t} = -\left[\widetilde{\mathbf{M}}_{I_1} \mid \mathbf{M}_{I_2}\right]\begin{bmatrix}\mathbf{d}\\ \hat{\mathbf{s}}_{I_2}\end{bmatrix} + \mathbf{s} \mod q.$$

The equation becomes an LWE problem if we view the secret vector as $\begin{bmatrix}\mathbf{d}\\ \hat{\mathbf{s}}_{I_2}\end{bmatrix}$ and the noise vector as $\mathbf{s}$.



(a) (b)

Fig. 1: Success rate on randomly generated uSVPγ in the lattice $\mathbf{B}_{\mathsf{HT}}$ for (a) 10 confirmed coefficients and (b) 20 confirmed coefficients

3.3 Hardness Analysis

Because of the special secret distribution of the secret vector $[\mathbf{d} \mid \hat{\mathbf{s}}_{I_2}]^\top$, both Kannan's and Bai-Galbraith's embedding cannot provide satisfying solution to the LWE problem. Instead we choose the half-twisted embedding to solve the instance which can re-balance the shortest vector in the lattice.

Following the construction of [WWTT18], we construct the basis of lattice by

$$\mathbf{B}_{\mathsf{HT}} = \left[\begin{array}{c|cc|c} \nu\mathbf{I}_{n_{I_1}} & & \widetilde{\mathbf{M}}_{I_1}^\top & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{I}_{n-n_{I_0}-n_{I_1}} & \mathbf{M}'_{I_2} & \mathbf{0} \\ & \mathbf{0} & q\mathbf{I}_{n_{I_0}+n_{I_1}} & \\ \hline \mathbf{0} & \mathbf{t}^\top & & \lambda \end{array}\right],$$

where $[\mathbf{I}_{n-n_{I_0}-n_{I_1}} \mid \mathbf{M}'_{I_2}]$ denotes the reduced row echelon matrix of $-\mathbf{M}_{I_2}^\top$. The lattice $\Lambda(\mathbf{B}_{\mathsf{HT}})$ contains a vector $[\nu\mathbf{d} \mid \mathbf{s} \mid \lambda]$. The distribution of $\mathbf{s}$ is the central binomial distribution β_η and $\mathbf{d}$ contains only $\{-1, 1\}$. Thus, to re-balance the vector, ν should be set to σ_s/σ_d where $\sigma_d = 1$ is the standard deviation of the vector $\mathbf{d}$. For Kyber512, $\sigma_s = \sqrt{6}/2$ and for Kyber768 or Kyber1024, $\sigma_s = 1$.

To determine the least number of coefficients we must recover in the CPA step, we perform an experiment on solving the uSVPγ randomly generated by script. The result is shown in Fig. 1, where the lines are the success rate of finding $[\nu\mathbf{d} \mid \mathbf{s} \mid \lambda]$ by the BKZ algorithm [2] of block size 50 for 20 randomly generated $\mathbf{s}$. The blue lines follow the same distribution of Kyber 768/1024 and the red lines follow that of Kyber512.

From the result, with 20 coefficients recovered in the NTT domain, or $n_{I_0} = 20$, we need $n_{I_1} \geq 35$ positive/negative coefficients in order to recover the secret key $\mathbf{s}$ of Kyber768/1024 and at $n_{I_1} \geq 43$ positive/negative coefficients for Kyber512. This is because Kyber512 has bigger standard deviation of $\mathbf{s}$ in its

[2] We ran the experiment using the BKZ implementation from `fpylll` in `Sage10.3`. See https://github.com/fplll/fpylll

specification, which results in a smaller gap between the shortest vector in $\mathbf{B}_{\mathsf{HT}}$. If we lower the recovered coefficients to $n_{I_0} = 10$, now we need at least $n_{I_1} \geq 62$ positive/negative coefficients for Kyber768/1024 and at least $n_{I_1} \geq 71$ positive/negative coefficients for Kyber512.

Notice that in order to do a full key recovery, the number of recovered coefficients and recovered positive/negative coefficients need to be multiplied by $2k$, where k is the module dimension for each version of Kyber.

4 Improved Power Analysis Attack on Masked Kyber

In this section, we present the result of applying our proposed approach to perform successful attacks on a protected implementation of Kyber. We experimented our attacks on simulated power traces of the ARM cortex-M0 processor, then estimate how many traces we need to conduct our attack.

4.1 Attack Target

We focus on the open-source and first-order masked implementation of Kyber from [BGR+21]. The polynomial arithmetic in the implementation is primarily written in assembly and has been adapted from the pqm4 project [KRSS19]. In the implementation, the central reduction algorithm is an improved Plantard reduction proposed in the paper [HZZ+22]. We generate our simulated traces using ELMO [ELM], a tool that emulates the power consumption of an ARM Cortex-M0 processor. This tool accurately reproduces the M0 processor's 3-stage pipeline, which means that the algorithmic noise is taken into account. The reliability of ELMO has been validated by comparing leakage detection results between simulated traces and real traces obtained from an STM32F0 Discovery Board [MOW17]. For reference, performing a successful key recovery power analysis on the lattice-based signature scheme FALCON requires 2000 simulated power traces and 5000 real traces [GMRR22].

4.2 Attack Detail

Our attack proceeds as follows: First, we perform a second-order CPA on Kyber to recover a portion of the coefficients in the secret key. We follow steps similar to Tosun et al. [TMS24], with the main difference being that we categorize the coefficients into three groups, as mentioned in Section 3.2. Next, we use the lattice attack described also in 3.2 to recover the remaining coefficients.

As given in [TMS24], the author proposed a range power model which is very effective against arithmetic masking when central reduction is employed. Consider any uniformly random variables $X \in \mathbb{Z}_q^{\pm} := [-q/2, q/2]$, the power model is defined as,

$$\mathcal{RN}_q(X) = \begin{cases} ((\beta/2)^2 - c_1)/c_0 & \text{if } X = 0 \\ |X|, & \text{otherwise} \end{cases}$$

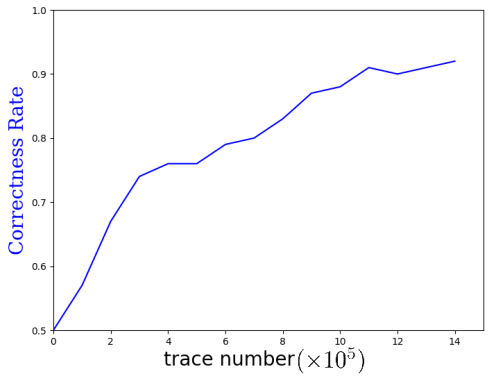


Fig. 2: Possibility of successful distinguishing $\hat{s}_i$ and $-\hat{s}_i$ in masked Kyber

where c_0, c_1 are some constant and β is the number of bits in the registers.

Notice that $(\hat{s}_{2i}, \hat{s}_{2i+1})$ is a degree-1 polynomial in Kyber, and two coefficients should be predicted together to mitigate the probability of false positives. We tested $q \cdot q/2$ hypotheses with $\mathcal{RN}_q$ as the prediction function, so that either the actual secret or its additive inverse is found. However, when $\mathcal{RN}_q$ is used as the prediction function, an hypothesis $(\hat{s}_{2i}, \hat{s}_{2i+1})$ and its additive inverse $(-\hat{s}_{2i}, -\hat{s}_{2i+1})$ will get the same correlation score due to the nature of absolute value function. Therefore, we cannot determine the sign of the recovered coefficients.

Since not every bit of the intermediate values equally contributes to the power consumption, the author of [TMS24] claims that the sign can be distinguished by re-running the same CPA attack on two hypotheses, $\pm\hat{s}_i$, using the sign function. Due to the absence of the trace in their online resource [3], we run our own experiment to verify the result. We notice that this technique works well only in the unmasked scenario. For masked Kyber, it achieves only about a 90% success rate, even when the number of traces exceeds 1 million, as shown in Figure 2. This is probably because the contribution of each bit is diluted when the secret intermediate value is split into random shares.

4.3 Evaluation

Figure 3 shows the number of recovered coefficients with second-order CPA using $\mathcal{RN}_q$ as the prediction function. We record the highest threshold that an incorrect coefficient pair can have for different numbers of traces, which is shown by the red line in the figure. Any coefficient pairs with a correlation coefficient higher than this threshold are thus confirmed to be correct, with ambiguity of the sign. From the analysis in Section 3.3, we need $n_{I_0} + n_{I_1}$ to be more than

[3] https://github.com/toluntosun21/ExploitingCentralReduction

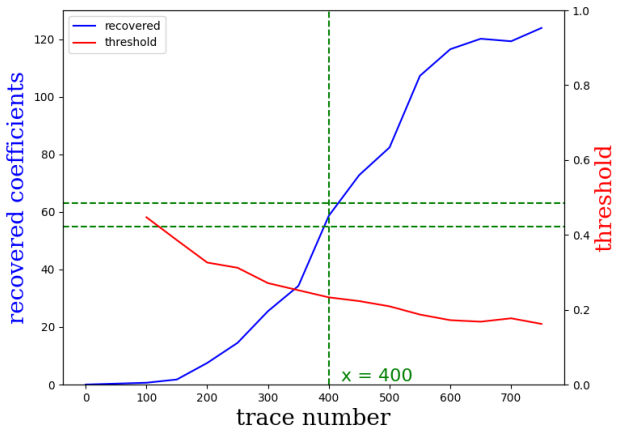


Fig. 3: Recovered coefficients and threshold for traces generated by ELMO

55, or 63 depending on the parameter set, to allow us to construct a solvable LWE instance for 20 confirmed coefficients. The 20 confirmed coefficients can be obtained by iterating through 2^{19} possibilities, and the reduction by one half is because a negative key is also accepted. From Figure 3, it can be seen that around 400 traces are enough to acquire enough coefficients to break the masked version of Kyber. If more traces are gathered, the complexity can also be reduced. For example, approximately 500 traces allow us collect enough coefficients to construct an LWE instance with $n_{I_0} = 10$, reducing the complexity to 2^9 iterations.

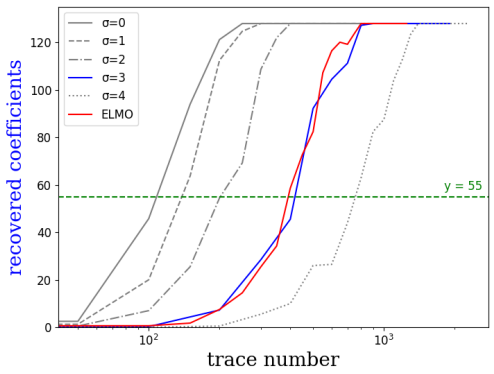


Fig. 4: Comparison between recovered coefficients of second-order CPA attacks on traces generated by Hamming weight model and traces generated by ELMO.

Figure 4 illustrates the CPA results across varying noise levels under the Hamming weight leakage model, with further discussion provided in the Appendix A.1. Note that the x-axis uses a logarithmic scale to display the number of traces. The results clearly show that the noise parameter σ has a significant influence on the attack's effectiveness. For instance, when $\sigma = 4$, approximately 800 traces are needed for successful key recovery, whereas only 110 traces are required when $\sigma = 0$. Additionally, we observed that traces generated by ELMO closely resemble those produced by the Hamming weight model at $\sigma = 3$, a finding that informs our experiments in the Appendix on other targets.

5 Conclusion

In this paper, we present a refined lattice analysis of the correlation power analysis attack on CRYSTALS-Kyber, resulting in a reduction of the required power traces for successful key recovery. We achieve this by introducing a novel coefficient classification algorithm based on adjustable thresholds. This algorithm categorizes the coefficients into confirmed, positive/negative, and unknown categories. Subsequently, we employ this information using the half-twisted embedding method to recover the secret key.

The experimental results validate the effectiveness of this refined approach. Through careful adjustment of threshold values, we successfully recovered the secret key using only 400 power traces. These advancements promise to enhance the understanding of vulnerabilities in cryptographic implementations and facilitate the development of more robust encryption techniques against such attacks.

Acknowledgment

This work is partially supported by JST CREST Grant Number JPMJCR2113, Japan, and JSPS KAKENHI Grant Number 25K21805, Japan.

Bibliography

[ABC+23] Melissa Azouaoui, Olivier Bronchain, Gaëtan Cassiers, Clément Hoffmann, Yulia Kuzovkova, Joost Renes, Tobias Schneider, Markus Schönauer, François-Xavier Standaert, and Christine van Vredendaal. Protecting Dilithium against leakage revisited sensitivity analysis and improved implementations. *IACR TCHES*, 2023(4):58–79, 2023.

[ACC+22] Amin Abdulrahman, Jiun-Peng Chen, Yu-Jia Chen, Vincent Hwang, Matthias J. Kannwischer, and Bo-Yin Yang. Multi-moduli NTTs for Saber on Cortex-M3 and Cortex-M4. *IACR TCHES*, 2022(1):127–151, 2022.

[BBE+18] Gilles Barthe, Sonia Belaïd, Thomas Espitau, Pierre-Alain Fouque, Benjamin Grégoire, Mélissa Rossi, and Mehdi Tibouchi. Masking the GLP lattice-based signature scheme at any order. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 354–384. Springer, Cham, April / May 2018.

[BDH+21] Shivam Bhasin, Jan-Pieter D'Anvers, Daniel Heinz, Thomas Pöppelmann, and Michiel Van Beirendonck. Attacking and defending masked polynomial comparison. *IACR TCHES*, 2021(3):334–359, 2021.

[BDK+18] Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS - kyber: A CCA-secure module-lattice-based KEM. In *2018 IEEE European Symposium on Security and Privacy*, pages 353–367. IEEE Computer Society Press, April 2018.

[BDK+21] Shi Bai, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Dilithium – algorithm specifications and supporting documentation (version 3.1), 2021. available at https://pq-crystals.org/dilithium/data/dilithium-specification-round3-20210208.pdf.

[BG14a] Shi Bai and Steven D. Galbraith. An improved compression technique for signatures based on learning with errors. In Josh Benaloh, editor, *CT-RSA 2014*, volume 8366 of *LNCS*, pages 28–47. Springer, Cham, February 2014.

[BG14b] Shi Bai and Steven D. Galbraith. Lattice decoding attacks on binary LWE. In Willy Susilo and Yi Mu, editors, *ACISP 14*, volume 8544 of *LNCS*, pages 322–337. Springer, Cham, July 2014.

[BGR+21] Joppe W. Bos, Marc Gourjon, Joost Renes, Tobias Schneider, and Christine van Vredendaal. Masking Kyber: First- and higher-order implementations. *IACR TCHES*, 2021(4):173–214, 2021.

[BPO+20] Florian Bache, Clara Paglialonga, Tobias Oder, Tobias Schneider, and Tim Güneysu. High-speed masking for polynomial comparison in lattice-based kems. *IACR TCHES*, 2020(3):483–507, 2020.

[CHK+21] Chi-Ming Marvin Chung, Vincent Hwang, Matthias J. Kannwischer, Gregor Seiler, Cheng-Jhih Shih, and Bo-Yin Yang. NTT multiplication for NTT-unfriendly rings. *IACR TCHES*, 2021(2):159–188, 2021.

[CKA+21] Zhaohui Chen, Emre Karabulut, Aydin Aysu, Yuan Ma, and Jiwu Jing. An efficient non-profiled side-channel attack on the CRYSTALS - dilithium post-quantum signature. In *Proc. ICCD 2021*, pages 583–590, 2021.

[CN11] Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better lattice security estimates. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 1–20. Springer, Berlin, Heidelberg, December 2011.

[CT65] James W. Cooley and John W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19:297–301, 1965.

[DHP+22] Jan-Pieter D'Anvers, Daniel Heinz, Peter Pessl, Michiel Van Beirendonck, and Ingrid Verbauwhede. Higher-order masked ciphertext comparison for lattice-based cryptography. *IACR TCHES*, 2022(2):115–139, 2022.

[DKL+18] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Dilithium: A lattice-based digital signature scheme. *IACR TCHES*, 2018(1):238–268, 2018.

[DKRV18] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, and Frederik Vercauteren. Saber: Module-LWR based key exchange, CPA-secure encryption and CCA-secure KEM. In Antoine Joux, Abderrahmane Nitaj, and Tajjeeddine Rachidi, editors, *AFRICACRYPT 18*, volume 10831 of *LNCS*, pages 282–305. Springer, Cham, May 2018.

[DVV23] Jan-Pieter D'Anvers, Michiel Van Beirendonck, and Ingrid Verbauwhede. Revisiting Higher-Order Masked Comparison for Lattice-Based Cryptography: Algorithms and Bit-Sliced Implementations . *IEEE Transactions on Computers*, 72(02):321–332, February 2023.

[ELM] ELMO: Evaluating leaks for the arm cortex-m0. https://github.com/sca-research/ELMO. Accessed: 2022-10-17.

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

[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, Berlin, Heidelberg, August 1999.

[GLP12] Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In Emmanuel Prouff and Patrick Schaumont, editors, *CHES 2012*, volume 7428 of *LNCS*, pages 530–547. Springer, Berlin, Heidelberg, September 2012.

[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 - 14th International Workshop, PQCrypto 2023*, pages 291–320. Springer, Cham, August 2023.

[GMRR22] Morgane Guerreau, Ange Martinelli, Thomas Ricosset, and Mélissa Rossi. The hidden parallelepiped is back again: Power analysis attacks on falcon. *IACR TCHES*, 2022(3):141–164, 2022.

[HHP+21] Mike Hamburg, Julius Hermelink, Robert Primas, Simona Samardjiska, Thomas Schamberger, Silvan Streit, Emanuele Strieder, and Christine van Vredendaal. Chosen ciphertext k-trace attacks on masked CCA2 secure Kyber. *IACR TCHES*, 2021(4):88–113, 2021.

[HNPS24] Julius Hermelink, Kai-Chun Ning, Richard Petri, and Emanuele Strieder. The insecurity of masked comparisons: SCAs on ML-KEM's FO-transform. In Bo Luo, Xiaojing Liao, Jun Xu, Engin Kirda, and David Lie, editors, *ACM CCS 2024*, pages 2430–2444. ACM Press, October 2024.

[HZZ+22] Junhao Huang, Jipeng Zhang, Haosong Zhao, Zhe Liu, Ray C. C. Cheung, Çetin Kaya Koç, and Donglong Chen. Improved plantard arithmetic for lattice-based cryptography. *IACR TCHES*, 2022(4):614–636, 2022.

[Kan87] Ravi Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12(3):415–440, 1987.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 388–397. Springer, Berlin, Heidelberg, August 1999.

[KO62] Anatolii Karatsuba and Yu Ofman. Multiplication of multidigit numbers on automata. *Soviet Physics Doklady*, 7:595, 1962.

[KRSS19] Matthias J. Kannwischer, Joost Rijneveld, Peter Schwabe, and Ko Stoffelen. pqm4: Testing and benchmarking NIST PQC on ARM Cortex-M4. Cryptology ePrint Archive, Report 2019/844, 2019.

[KT23] Yen-Ting Kuo and Atsushi Takayasu. A lattice attack on CRYSTALS-kyber with correlation power analysis. In Hwajeong Seo and Suhri Kim, editors, *ICISC 23, Part I*, volume 14561 of *LNCS*, pages 202–220. Springer, Singapore, November / December 2023.

[LLL82] H.W. jr. Lenstra, A.K. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:515–534, 1982.

[LLZ+24] Yong Liu, Yuejun Liu, Yongbin Zhou, Yiwen Gao, Zehua Qiao, and Huaxin Wang. A novel power analysis attack against crystals-dilithium implementation. In *2024 IEEE European Test Symposium (ETS)*, pages 1–6, 2024.

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23. Springer, Berlin, Heidelberg, May / June 2010.

[MGTF19] Vincent Migliore, Benoît Gérard, Mehdi Tibouchi, and Pierre-Alain Fouque. Masking Dilithium - efficient implementation and side-channel evaluation. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *ACNS 19International Conference on Applied Cryptography and Network Security*, volume 11464 of *LNCS*, pages 344–362. Springer, Cham, June 2019.

[Mil86] Victor S. Miller. Use of elliptic curves in cryptography. In Hugh C. Williams, editor, *CRYPTO'85*, volume 218 of *LNCS*, pages 417–426. Springer, Berlin, Heidelberg, August 1986.

[MOW17] David McCann, Elisabeth Oswald, and Carolyn Whitnall. Towards practical tools for side channel aware software engineering: 'grey box' modelling for instruction leakages. In Engin Kirda and Thomas Ristenpart, editors, *USENIX Security 2017*, pages 199–216. USENIX Association, August 2017.

[MWK+24] Catinca Mujdei, Lennert Wouters, Angshuman Karmakar, Arthur Beckers, Jose Maria Bermudo Mera, and Ingrid Verbauwhede. Side-channel analysis of lattice-based post-quantum cryptography: Exploiting polynomial multiplication. *ACM Trans. Embed. Comput. Syst.*, 23(2), March 2024.

[NDGJ21] Kalle Ngo, Elena Dubrova, Qian Guo, and Thomas Johansson. A side-channel attack on a masked IND-CCA secure Saber KEM implementation. *IACR TCHES*, 2021(4):676–707, 2021.

[NDJ23] Kalle Ngo, Elena Dubrova, and Thomas Johansson. A side-channel attack on a masked and shuffled software implementation of Saber. *Journal of Cryptographic Engineering*, 13(4):443–460, November 2023.

[NIS24] Module-lattice-based key-encapsulation mechanism standard. National Institute of Standards and Technology, NIST FIPS PUB 203, U.S. Department of Commerce, August 2024.

[OSPG18] Tobias Oder, Tobias Schneider, Thomas Pöppelmann, and Tim Güneysu. Practical CCA2-secure masked Ring-LWE implementations. *IACR TCHES*, 2018(1):142–174, 2018.

[OY23] Sıla Özeren and Oğuz Yayla. Methods for masking crystals-kyber against side-channel attacks. In *2023 16th International Conference on Information Security and Cryptology (ISCTürkiye)*, pages 1–6, 2023.

[PP19] Peter Pessl and Robert Primas. More practical single-trace attacks on the number theoretic transform. In Peter Schwabe and Nicolas

Thériault, editors, *LATINCRYPT 2019*, volume 11774 of *LNCS*, pages 130–149. Springer, Cham, October 2019.

[PPM17] Robert Primas, Peter Pessl, and Stefan Mangard. Single-trace side-channel attacks on masked lattice-based encryption. In Wieland Fischer and Naofumi Homma, editors, *CHES 2017*, volume 10529 of *LNCS*, pages 513–533. Springer, Cham, September 2017.

[PRB09] Emmanuel Prouff, Matthieu Rivain, and Regis Bevan. Statistical analysis of second order differential power analysis. *IEEE Transactions on Computers*, 58(6):799–811, 2009.

[QCD19] Yue Qin, Chi Cheng, and Jintai Ding. An efficient key mismatch attack on the NIST second round candidate Kyber. Cryptology ePrint Archive, Report 2019/1343, 2019.

[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, *ASIACRYPT 2021, Part IV*, volume 13093 of *LNCS*, pages 92–121. Springer, Cham, December 2021.

[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.

[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.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the Association for Computing Machinery*, 21(2):120–126, February 1978.

[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.

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th FOCS*, pages 124–134. IEEE Computer Society Press, November 1994.

[SVO+10] François-Xavier Standaert, Nicolas Veyrat-Charvillon, Elisabeth Oswald, Benedikt Gierlichs, Marcel Medwed, Markus Kasper, and Stefan Mangard. The world is not enough: Another look on second-order DPA. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 112–129. Springer, Berlin, Heidelberg, December 2010.

[THM+07] Michael Tunstall, Neil Hanley, Robert P. McEvoy, Claire Whelan, Colin C. Murphy, and William P. Marnane. Correlation power analysis of large word sizes. In *IETIrish Signals and Systems Conf-ISSC*, page 145–150, 2007.

[TMS24] Tolun Tosun, Amir Moradi, and Erkay Savas. Exploiting the central reduction in lattice-based cryptography. *IEEE Access*, 12:166814–166833, 2024.

[Too63] A.L Toom. The complexity of a scheme of functional elements realizing the multiplication of integers. In *Soviet Mathematics-Doklady*, volume 3, pages 714–716, 1963.

[TS24] Tolun Tosun and Erkay Savas. Zero-value filtering for accelerating non-profiled side-channel attack on incomplete ntt-based implementations of lattice-based cryptography. *IEEE Transactions on Information Forensics and Security*, 19:3353–3365, 2024.

[WWTT18] Weiyao Wang, Yuntao Wang, Atsushi Takayasu, and Tsuyoshi Takagi. Estimated cost for solving generalized learning with errors problem via embedding techniques. In Atsuo Inomata and Kan Yasuda, editors, *IWSEC 18*, volume 11049 of *LNCS*, pages 87–103. Springer, Cham, September 2018.

[WXT24] Kai Wang, Dejun Xu, and Jing Tian. An improved two-step attack on kyber, 2024. available at https://arxiv.org/abs/2407.06942.

A Appendix: Other Applications

In this appendix, we extend the applicability of our proposed power analysis attack to other post-quantum cryptosystems that also utilize NTT-based polynomial multiplication. We demonstrate how our technique can be successfully migrated to first-order masked implementations of SABER and Dilithium, providing a detailed breakdown of the attack outline, leakage models, and experimental results for each.

A.1 Application to masked SABER

SABER [DKRV18] is a finalist in the NIST Post-Quantum Cryptography standardization process. Its security is built on the Module Learning with Rounding (Module-LWR) problem, which eliminates the need for error sampling by leveraging rounding for noise generation. The polynomial ring used within Saber is $R_q = \mathbb{Z}_q[x]/(x^n + 1)$ with $q = 2^{13}$ and $n = 256$ across all parameter sets. Saber also offers three security levels: Lightsaber with security level similar to AES-128, Saber with one similar to AES-192 and Firesaber with one similar to AES-256.

SABER was not originally designed to take advantage of NTT-friendly polynomial multiplication. Instead, it uses Toom-Cook-4 [Too63] and Karatsuba [KO62] for efficient arithmetic. However, Chung *et al.* [CHK+21] showed that NTTs could be adapted for SABER by choosing a large prime modulus $q' > nq^2/2$ with $n|(q' - 1)$, performing multiplications in $\mathbb{Z}_{q'}[x]$, and reducing back to $\mathbb{Z}_q[x]$. This adaptation impacts side-channel analysis by requiring attacks to target smaller portions of intermediate values, reducing the signal-to-noise ratio.

Furthermore, Abdulrahman *et al.* [ACC+22] addresses applying the NTT to SABER with masking for side-channel resistance. To secure SABER's polynomial arithmetic, they adapt multi-moduli NTT techniques that split computations across multiple moduli using Chinese remainder theorem. These allow efficient handling of masked shares while minimizing memory overhead. The implementation optimizes arithmetic operations to achieve side-channel security on embedded platforms like Cortex-M3 and Cortex-M4. Their speed-optimized masked M4 implementation is 16% faster than the fastest masked implementation using Toom–Cook.

As secret polynomials $\mathbf{s}$ and $\mathbf{s}'$ are masked arithmetically modulo q, the product can be larger than 32-bit. This implies switching to an NTT-friendly 25-bit modulus and performing 32-bit NTTs no longer produces correct results. To manage this, they combined a 32-bit NTT with a 16-bit NTT to compute the 48-bit value and then reduce each coefficient to $\mathbb{Z}_q$. The 32-bit NTT and 16-bit NTT is done by choosing $p_0 = 44683393$ and $p_1 = 769$ as moduli. Their product $q' = p_0 p_1 = 44683393 \cdot 769 = 34361529217 > 34359738368 = 2 \cdot (\frac{q}{2})^2 \cdot 256 \cdot 4$ shows that after applying CRT, the result is correct in $\mathbb{Z}$.

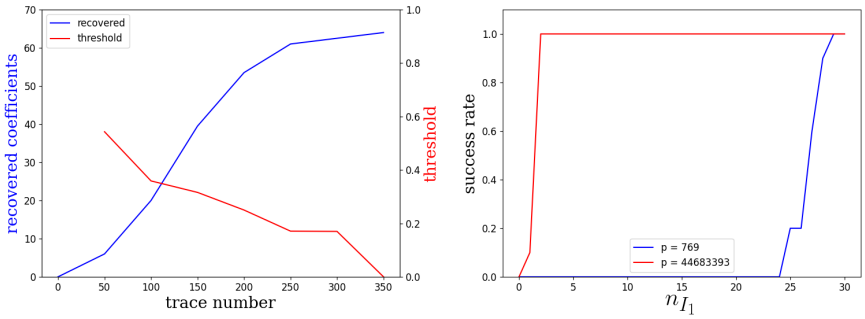
Leakage Model. We define the leakage function of the target device based on the random variable X, constant scaling factor α, and noise sampled from a Gaussian distribution with mean μ and standard deviation σ, which is independent of X. The leakage $\mathcal{L}(X)$ is defined as

$$\mathcal{L}(X) = \alpha \cdot HW(X) + \mathcal{N}(\mu, \sigma).$$

and it is commonly used to simulate SCA leakage of processors in the presence of noise when X is processed. For the following experiment, we use parameters $\alpha = 1$, $\mu = 0$, and $\sigma = 3$, as these values best align with the observed results of our Kyber experiment.

Attack Outline. We attack on the SABER variant where the security level is the same as Kyber-768. Consider the base multiplication $s \times u$ where $s \in \mathbb{R}_q$ is a secret polynomial, and $u \in \mathbb{R}_q$ is a public polynomial. The multiplication is split into 2 NTT-based multiplication in the ring $\mathbb{Z}_{p_0}[x]$ and $\mathbb{Z}_{p_1}[x]$. We focus on the NTT in the ring $\mathbb{Z}_{p1}[x]$ since it will be enough to determine the correct key. Since the implementation from above uses 6 layers of NTT, we divide the coefficients into $256/2^6 = 4$ groups and find the minimum number of coefficients we needed to recover other ones. The rest of the attack is similar to the attack on masked Kyber. First we compute the correlation coefficients of leakage with the power model proposed by [TMS24]. Then choose the coefficients whose correlation coefficient is higher than some threshold, which is determined by the maximum correlation coefficient of incorrect ones. Finally, use these coefficients to construct the LWE in section 3.2, while guessing the sign of first 10 coefficients.

Experimental Result. Figure 5 shows the result of our lattice attack and power analysis on masked SABER. You can see that it needs 29 coefficients out



(a) (b)

Fig. 5: (a) Recovered coefficients and threshold and (b) success rate on randomly generated uSVP$_\gamma$ when $n_{I_0} = 10$ for masked SABER

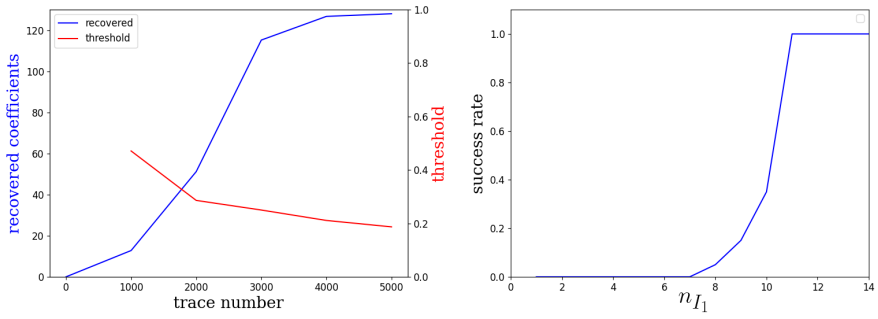
of 64 to guarantee a successful attack on masked Saber, which means a total of $29 \cdot 4 \cdot 3 = 348$ coefficients are needed. Combine with the power analysis result, approximately 150 traces are needed to perform a full-key recovery on this implementation. From the graph, it is also known that to attack the NTT in ring $\mathbb{Z}_{p_0}[x]$, only 11 out of 64 coefficients are needed to perform full-key recovery. However, the power analysis part of the attack is quite infeasible in ring $\mathbb{Z}_{p_0}[x]$, since it requires an enumeration of $p_0/2$ coefficients, which is significantly slower than the $p_1/2$ in the other ring. Therefore, we could reduce the number of traces even more by attacking NTT in the ring $\mathbb{Z}_{p_0}[x]$ for the cost of computation time.

A.2 Application to masked Dilithium

Dilithium [DKL+18] is a lattice-based digital signature scheme designed for post-quantum security, leveraging the hardness of the Module-LWE and Module Short Integer Solution (Module-SIS) problems. Dilithium's design emphasizes efficiency, compactness, and scalability across various parameter sets, making it suitable for diverse applications. Its cryptographic operations rely heavily on polynomial arithmetic over the ring $\mathbb{Z}_q[x]/(x^n + 1)$, where q is a prime modulus, and n is a power of two.

The first instance of high-order masking applied to a lattice-based signature scheme was introduced in [BBE+18], targeting the GLP signature scheme [GLP12]. Dilithium, an advanced version of GLP, incorporates compression methods from [BG14a] and various optimizations. In [MGTF19], high-order masking for Dilithium was implemented, along with a simplified version that employs a power-of-two modulus q instead of a prime modulus. This adjustment significantly enhanced the performance of their countermeasure, making it more efficient for practical implementations.

In a signature generation algorithm, not all variables necessarily require masking. For Dilithium, the authors of [ABC+23] revisited the sensitivity anal-



(a) (b)

Fig. 6: (a) Recovered coefficients and threshold and (b) success rate on randomly generated uSVP$_\gamma$ when $n_{I_0} = 10$ for masked Dilithium

ysis presented in [MGTF19] and identified intermediate variables that were incorrectly left unmasked in [MGTF19], potentially exposing the private key. Conversely, they also demonstrated that some variables in [MGTF19] were unnecessarily masked. Specifically, they argued that the vector $\mathbf{w} = \mathbf{Ay}$ must be masked. If left unmasked, since $\mathbf{A}$ is either a square matrix or has more rows than columns, the variable $\mathbf{y}$ can be efficiently derived from $\mathbf{w}$, which would enable recovery of the secret key. On the other hand, the variable $\mathbf{r} = \mathbf{w} - c\mathbf{s}_2 = \mathbf{Az} - c\mathbf{t}$ can be treated as public after rejection sampling. Thus, the computation of the hint vector $\mathbf{h}$ can be performed without masking. In the following attack, we target the same masking strategy as [ABC+23].

Attack Outline. We target the NIST security level II implementation and use the same leakage model as the case of SABER. Since the computation of $\mathbf{h}$ is unmasked, our attack focuses on the computation of $\mathbf{z} = \mathbf{y} + c\mathbf{s}_1$, particularly on the multiplication $c \cdot \mathbf{s}_1$. Similar to Kyber, Dilithium employs point-wise operations with private keys in the NTT domain. The reference implementation of $\hat{c} \cdot \hat{\mathbf{s}}_1$ in Dilithium also uses a 7-layer NTT, as discussed in Section 2.4. While performing SCAs on these operations using CPA is theoretically similar to the approach for Kyber and SABER, the process becomes significantly more challenging in the case of Dilithium. This is due to its modulus $q = 8380417$, which makes exhaustive enumeration over the entire key space computationally infeasible.

Tunstall et al. [THM+07] demonstrated the application of CPA to attack cryptographic implementations using large word sizes, successfully extracting a DES key on a 32-bit platform. Their approach involves dividing large intermediate byte values into smaller segments and performing CPA on each segment independently. For an intermediate value $f(x_1, \ldots, x_p, k)$, where k is an l-bit unknown fixed value and x_i represents known changing inputs, k is divided into consecutive blocks $b_{n-1}, \ldots, b_0$ from the most significant to the least significant

bits, with each block containing l_{b_i} bits. The Pearson correlation coefficient for each block is then scaled with a factor of $\sqrt{l_{b_i}/l}$.

The practical attack procedure is as follows: First, calculate the PCCs for all possible values of b_0, rank them in descending order, and select the top h_0 candidates. Secondly, Combine the h_0 candidates for b_0 with b_1 to generate a new set of candidates. Compute the PCCs for this set, rank them in descending order, and select the top h_1 candidates. Repeat step 2 recursively for the subsequent blocks b_i $(2 \leq i < n-2)$. And finally, merge the h_{n-2} candidates for b_{n-2} with b_{n-1}, compute the PCCs for all possible combinations, and select the candidate with the highest PCC as the final result.

The rest of the attack is similar to the attack on masked Kyber. We use these coefficients to construct the LWE in section 3.2, while guessing the sign of first 10 coefficients.

Experimental Result. Figure 6 illustrates the outcome of our lattice attack and power analysis on masked Dilithium. It shows that 11 coefficients out of 128 are required to ensure a successful attack on masked Dilithium, meaning a total of $11 \cdot 2 \cdot 4 = 88$ coefficients are needed. Coupled with the power analysis results, approximately 1000 traces are necessary to achieve full-key recovery on this implementation, which is only 20% of the traces required to recover all coefficients.