Exploiting the Central Reduction in Lattice-Based Cryptography

Tolun Tosun[1,3], Amir Moradi[2] and Erkay Savas[1]

[1] Sabanci University, Istanbul, Türkiye
{toluntosun,erkays}@sabanciuniv.edu
[2] Technische Universität Darsmtadt, Darmstadt, Germany
amir.moradi@tu-darmstadt.de
[3] Analog Devices

Abstract. This paper presents a novel and efficient way of exploiting side-channel leakage of masked implementations of lattice-based cryptography (LBC). The presented attack specifically targets the **central reduction** technique, which is widely adapted in efficient implementations of LBC. We show that the central reduction leads to a vulnerability by creating a strong dependency between the power consumption and the sign of sensitive intermediate variables. We exploit this dependency by introducing a novel hypothetical power model, the **range power model**, which can be employed in higher-order multi-query side-channel analysis attacks. We particularly show that our approach is valid for the prime moduli employed by Kyber and Dilithium, the lattice-based post-quantum algorithms selected by NIST, while it generalizes to other primes used in LBC as well. We practically evaluate our introduced approach by performing second-order non-profiled attacks against a masked implementation of Kyber on an Arm Cortex-M4 micro-processor. In our experiments we revealed the full secret key of the aforementioned implementation with only **2100** electro-magnetic (EM) traces without profiling, achieving a more than 14 times reduction in the number of traces compared to classical attacks.

Keywords: Side-Channel Analysis · Correlation Power Analysis · Post-Quantum Cryptography · Kyber · Dilithium · Plantard · Montgomery · Arithmetic Masking · Centered Reduction

1 Introduction

Shor's algorithm [Sho94] violates the security of traditional public-key cryptography including RSA and ECC through quantum computing. As the development of a larger quantum computer in the number of qubits is being reported each year, the quantum threat gradually becomes a reality. On the other hand, NIST's post-quantum cryptography contest is in the fourth round, with already selected algorithms. Among the winners, the lattice-based algorithms form the majority: Kyber [SAB+22], Dilithium [LDK+22] and Falcon [PFH+22]. Although the post-quantum algorithms can resist quantum computing attacks, special attention should be paid to side-channel analysis attacks [KJJ99, BCO04] when implementing these algorithms in both hardware and software.

The core operation in lattice-based cryptography (LBC) is the polynomial multiplication. For an efficient implementation, the Number Theoretic Transform (NTT) stands out as an excellent approach. NTT is indeed a special form of the Fast Fourier Transform (FFT) that operates on a discrete space. An important building block that significantly impacts the efficiency of the NTT algorithm is the modular reduction of integers, concerning the arithmetic for coefficients of polynomials. Classical techniques such as the Montgomery

reduction [Mon85] and Barrett reduction [Bar87] are already applied to LBC by the existing literature [AHKS22, GKS20, ABCG20, BKS19, Sei18]. The same holds for the the relatively new method the Plantard reduction [Pla21, HZZ+22]. One important distinction regarding the integer reduction in LBC compared to the RSA and ECC is the bit-length of the number to be reduced. LBC requires a reduction of relatively smaller numbers, that usually fit into a single computer word. For instance, Kyber employs a 11-bit coefficient modulus and Dilithium a 23-bit one. Moreover, the signed representation of integers over a modulus instead of the classical unsigned representation is more desired in LBC [HZZ+22, AHKS22, GKS20, ABCG20, BKS19]. That is to make a central reduction (a.k.a *centered* reduction) to the range $[-q/2, q/2]$ instead of $[0, q)$ for an odd modulus q. The Plantard and Montomgery algorithms enable 2-cycle implementation of central reduction on the Arm Cortex-M4 [HZZ+22, GKS20, ABCG20] while Plantard is superior since the output of Montgomery reduction requires a final subtraction or addition to be correct.

Main Motivation. Overall, the central reduction improves the efficiency of LBC implementations; however, it also creates new directions for side-channel analysis attacks. A well-known fact regarding the CMOS circuits is that the power consumption has a relatively strong dependency on the Hamming weight of the processed data. Respectively, the sign of a number in $[-q/2, q/2]$ becomes the dominant factor influencing its power consumption considering 2's complement to represent negative numbers. Based on this main motivation, we explore the characteristics of the central reduction in terms of side-channel leakage and exploit them, particularly in the presence of the masking countermeasure.

Related Work on Masking of LBC. It is well known that masking countermeasures provide a promising way of mitigating EM/power based side-channel analysis attacks. Accordingly, the existing literature on masking LBC is already quite rich. Some examples applied on Kyber [HKL+22, BGR+21a, FBR+21, ÖY23], Dilithium [MGTF19, ABC+23b, CGTZ23], Saber (another promising post-quantum lattice-based KEM) [BDK+20, KDVB+22] and generic lattice-based encryption [RRd+16, BC22, CGMZ21]. Indeed, masking the polynomial arithmetic is considered trivial and is achieved by simply repeating the operations. For instance, a polynomial multiplication $a \cdot b$ can be performed through the random shares a^0 and a^1 individually while satisfy $a = a^0 + a^1$, instead of accessing a in plain. On the other hand, masking the non-linear components of algorithms involves specialized techniques. One simple example is the compression operation in Kyber, which aims to identify the interval each coefficient of a given polynomial a resides. It is easy to see that such an operation cannot be performed independently on the arithmetic shares, as it is the case with polynomial multiplication. A masked implementation of such non-linear operations typically utilizes arithmetic-to-Boolean mask conversion [RRd+16, BGR+21b, FBR+22, BC22, KDVB+22, BDK+20, HKL+22] while a Boolean masking scheme ensures $a = a^0 \oplus a^1$. Overall, the existing work regarding masking LBC aims to achieve provable first- or higher-order security by introducing more efficient solutions for masking the non-trivial parts of the algorithms. Particularly, current masked implementations on the Cortex-M4 [HKL+22, BGR+21a, ABC+23b, BDK+20, HDR23] inherit the polynomial arithmetic from the well-known pqm4 library [KRSS19a], known for providing state-of-the-art yet unprotected implementations of post-quantum algorithms. To the best of our knowledge, there has been no work questioning the difficulty of possible attacks on the linear parts as long as the implementations are proven to be secure in the desired security order.

Related Work on Attacks against LBC. On the other hand, there exist many side-channel analysis attacks in the literature that target implementations of LBC. These works

can be considered in two classes: profiled attacks such as [PPM17, KLH+20, XPRO20, BAE+23, BNGD22, DNG22, MUTS22, KAA22, AAT+21] and non-profiled attacks such as [MBB+22, CKA+21, TS23, SLKG22]. The non-profiled attacks commonly target polynomial multiplication for which a secret polynomial is multiplied with a publicly known polynomial that changes based on the input given to the victim implementation. In [MBB+22], the authors show that the performance of non-profiled attacks against the polynomial multiplication directly depends on the employed multiplication algorithm as well as the parameters such as the coefficient modulus. While for some instances of LBC, the non-profiled attack to retrieve secret polynomials can take a relatively long time, acceleration is possible in certain scenarios by collecting more measurements as shown in [CKA+21, TS23] for distinct implementations. The authors of [TS23] particularly focused on so-called *incomplete* NTT, a special case of NTT that is employed in efficient implementations of both Kyber and Dilithium on the Cortex-M4 [KRSS19a]. Besides, it is shown in [SLKG22] that the polynomial multiplication can be also effectively targeted in the case of hardware implementations. Except [TS23], the aforementioned non-profiled attacks target unprotected implementations, *i.e.* not masked. On the other hand, profiled attacks demand for a stronger adversary model. Particularly, an open device for profiling that is identical to the victim must be available. However, the majority of side-channel analysis attacks published on LBC make use of a profiling device. Several operations of LBC have been selected as the target of the attacks. Among them, [PPM17, XPRO20, KLH+20] focus on the NTT transformation. The attack presented in [PPM17] is distinctive by revealing *single-trace* vulnerabilities of LBC. It should be emphasized that single-trace profiling attacks cannot be avoided by masking, since side-channel leakage of a single execution is measured, which is not affected by the randomization introduced by masking. The authors of [BAE+23] specifically focuses on the multiplication of polynomials with small coefficients. The works [BNGD22, DNG22, MUTS22] present attacks on encoding/decoding functions that transform binary input into a polynomial or vice versa. We should note that [DNG22] presents only a message recovery attack that aims to retrieve the decapsulated message rather than the secret key. The target of [KAA22] is the sampling of challenge polynomials, which have a limited number of non-zero coefficients that are also small in magnitude. Table 1 formalizes the above discussion and positions our study among the existing attacks from the literature.

Our Contributions. What follows is a list of the contributions we have made to this work.

- To the best of our knowledge, we present the first study in the literature that is particularly developed for and effective against side-channel protected implementations of LBC without the need for profiling.
- We show that the central reduction techniques that are widely adapted in LBC lead to a vulnerability from side-channel analysis perspective. Particularly, information about the sign of arithmetic shares would ease exploiting the leakage and conducting successful key-recovery attacks.
- We show that the employed coefficient modulus as well as the reduction algorithm affect the side-channel leakage of masked implementations of LBC, particularly making non-profiled attacks easier to conduct.
- We introduce a novel hypothetical power model for non-profiled side-channel analysis attacks, namely the range power model that exploits the vulnerability caused by the adaption of central reduction in masked implementations of LBC.
- We apply the range power model against a first-order masked implementation of the lattice-based post-quantum KEM Kyber. We further experimentally show that our

Table 1: Qualitative summary of state-of-the-art side-channel analysis attacks on implementations of LBC.

Work	Class	Algorithm	Implementation	Masked	Target Function
this work	Non-Profiled	Dilithium$^{\text{☆}}$, Kyber	Cortex-M4$^{\bigstar}$	✓✓	poly. mult.
[MBB+22]	Non-Profiled	Kyber, Saber, NTRU[1]	Cortex-M4$^{\bigstar}$	✗	poly. mult.
[CKA+21]	Non-Profiled	Dilithium	Ref. C	✗	poly. mult.
[TS23]	Non-Profiled	Dilithium, Kyber	Cortex-M4$^{\bigstar}$	✓	poly. mult.
[SLKG22]	Non-Profiled	Dilithium	Hardware	✗	poly. mult.
[PPM17]	Profiled	✈	Cortex-M4$^{\text{✧}}$	✓✓	NTT
[KLH+20]	Profiled	Dilithium	Ref. C	✗	NTT
[KLH+20]	Profiled	Dilithium	Ref. C*	✓	sparse poly. mult.
[XPRO20]	Profiled	Kyber	Ref. C	✗	NTT
[XPRO20]$^{\text{✉}}$	Profiled	Kyber	Cortex-M4$^{\bigstar}$	✗	bin. to poly.
[BAE+23]	Profiled	Dilithium	✤	✗	small poly.mult
[BNGD22]	Profiled	Kyber, Saber	Cortex-M4$^{\text{❧}}$	✓✓	poly. to bin.
[DNG22]$^{\text{✉}}$	Profiled	Kyber	Cortex-M4$^{\text{❧}}$	✓	bin. to poly.
[MUTS22]	Profiled	Dilithium	Ref. C	✗	bin. to poly.
[KAA22]$^{\text{✉}}$	Profiled	Dilithium, NTRU[1] NTRU Prime[2]	Ref. C Cortex-M4$^{\bigstar}$	✓	small poly. sampling

☆ attacks in the simulation
★ from [KRSS19b]
✈ generic to NTT applications in LBC. ✧ from [RRd+16]
✓✓ presents a novel technique to tackle masking
✶ attacks through an implementation submitted for another project [BAA+19]
✤ not reported
✉ challenge polynomial/message-recovery attack
❧ from [HKL+22]
[1] a post-quantum lattice-based KEM [CDH+20]
[2] a post-quantum lattice-based KEM [BBC+20]

approach reduces the number of traces required for a successful attack by an order of magnitude compared to the existing and well-known approaches.

2 Notations

The notations we followed in the paper are as follows.

- Vectors are represented by bold lowercase letters such as $\mathbf{a}$, while matrices are represented by bold uppercase letters such as $\mathbf{A}$. Polynomials are represented by lowercase regular letters such as a. Vector-to-vector, matrix-to-vector, and scalar-vector multiplications are denoted by $\cdot$ while element-wise multiplication of vectors or matrices is denoted by $\star$. The i-th element (coefficient) of a vector (polynomial) is denoted by the subscripts, such as $\mathbf{a}_i$ (a_i).
- The central reduction to the range $[-q/2, q/2]$ is explicitly denoted by $\bmod^{\pm} q$ (assuming an odd q), while $\bmod q$ denotes the regular modular reduction to the range $[0, q)$. We also use $\bmod q$ when the output range is not important such as in high-level representation of algorithms, *i.e.* pseudocodes. The set of unsigned integers in $[0, q)$ is denoted by $\mathbb{Z}_q$. Accordingly, ${}^{\pm}\mathbb{Z}_q$ represents the signed representation of integers modulo q, namely the integers in the range $[-q/2, q/2]$. We assume an odd q unless the opposite is explicitly stated.
- Random variables are denoted by uppercase letters such as X. $\mathcal{P}(\cdot)$ denotes the probability function. $E[\cdot]$ denotes the expected value function while the sample mean over a random variable is denoted by the overbar, such as $\overline{X}$. $\mathcal{N}(\mu, \sigma)$ denotes the noise following a Gaussian distribution with mean μ and standard deviation σ.
- Unless otherwise stated, the logarithm is base 2, and $\oplus$ denotes the exclusive OR.
- Shares of variables are represented by superscripts, such as $X = X^0 + X^1$.
- $\mathcal{B}(X)$ denotes the number of bits needed to represent the unsigned integer X. β denotes the machine word size. In this paper, either $\beta = 16$ or $\beta = 32$.
- $\mathcal{HW}_\beta(X)$ represents the Hamming weight of a signed integer X in β-bit 2's complement representation.
- $\mathcal{S}(X) : \mathbb{Z} \rightarrow \{0, 1\}$ returns the non-negativeness (sign) of the integer X:

$$\mathcal{S}(X) = \begin{cases} 1, & \text{if } X \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

- The outcome of the sign equality check between the variables X and Y is denoted by $\mathcal{I}(X, Y)$:

$$\mathcal{I}(X, Y) = \begin{cases} 1, & \text{if } \mathcal{S}(X) = \mathcal{S}(Y) \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

3 Lattice-Based Cryptography (LBC)

In this section, we briefly review lattice-based post-quantum algorithms from a side-channel analysis attack perspective. We focus on Kyber [SAB+22] and Dilithum [LDK+22], which are among the algorithms selected by NIST at the end of the third round of the post-quantum cryptography standardization process. Afterwards, we discuss the details of the NTT, which is a crucial primitive for efficiently implementing polynomial arithmetic in LBC.

Ring of Polynomials. Most of the lattice-based cryptosystems, including Kyber and Dilithium, operate over the ring of polynomials $\mathcal{R}_q = \mathbb{Z}_q[x]/(X^n + 1)$ that contains polynomials up to degree $n-1$ while coefficients are in $\mathbb{Z}_q$. Arithmetic operations in $\mathcal{R}_q$ are the main building block for implementing LBC, and it is the main target of side-channel analysis attacks as well.

Kyber. Kyber [SAB+22] is a lattice-based post-quantum KEM, a variant of the LPR encryption scheme [LPR10]. For all security levels, Kyber employs $q = 3329$ and $n = 256$ to instantiate $\mathcal{R}_q$. The key pair is generated by the MLWE [Reg05] equation, $\mathbf{t} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$. The vector of polynomials $\mathbf{e} \in \mathcal{R}_q^k$ is considered as noise and thrown away after the key generation while $\mathbf{s} \in \mathcal{R}_q^k$ forms the secret key. On the other hand, $\mathbf{A} \in \mathcal{R}_q^{k \times k}$ and $\mathbf{t} \in \mathcal{R}_q^k$ are public. The polynomials in both $\mathbf{s}$ and $\mathbf{e}$ are *short*, whose coefficients are sampled from the central binomial distribution $\mathcal{CBD}_\eta$. The sensitive operation in a KEM in terms of non-profiled side-channel analysis attacks is the decapsulation function as the secret key is involved [MBB+22, TS23]. The decapsulation in an LPR scheme such as Kyber is quite simple: $v - \mathbf{s}^T \cdot \mathbf{u}$, where $\mathbf{u} \in \mathcal{R}_q^k$ and $v \in \mathcal{R}$ together form the ciphertext. Related parameters k and η are chosen depending on the NIST security level as $\{2, 3, 4\}$ and $\{2, 4, 2\}$, respectively.

Dilithium. Dilithium is a lattice-based post-quantum signature following the Fiat-Shamir scheme with aborts approach [Lyu09]. It employs $q = 2^{23} - 2^{13} + 1 = 8380417$ and $n = 256$. The secret-public key pair for Dilithium is generated through the MLWE equation similar to Kyber, *i.e.* $\mathbf{t} = \mathbf{A} \cdot \mathbf{s}_1 + \mathbf{s}_2$. Distinctively, both $\mathbf{s}_1$ and $\mathbf{s}_2$ are saved as the secret key while the pseudo-randomly generated matrix $\mathbf{A} \in \mathcal{R}_q^{k \times l}$ and $\mathbf{t} \in \mathcal{R}_q^l$ are public as in Kyber. Also, the coefficients of the secret polynomials in $\mathbf{s}_1$ and $\mathbf{s}_2$ are short as well. Specifically, the secret coefficients are sampled uniformly at random in $[-\eta, \eta]$. A natural target for a non-profiled side-channel analysis attack on Dilithium is the signature function as it involves the secret key [CKA+21, SLKG23, TS23]. More precisely, the multiplications $c \cdot \mathbf{s}_1$ and $c \cdot \mathbf{s}_2$ are targeted, where – among the outputs of the signature – the challenge polynomial $c \in \mathcal{R}_q$ is public and depends on the input message. The parameters (k, l) are chosen as $\{(4, 4), (6, 5), (8, 7)\}$ with respect to the security level. η is chosen the same as Kyber.

3.1 Number Theoretic Transform (NTT)

NTT allows efficient multiplication of polynomials in $\mathcal{R}_q$. Given two polynomials $a \in \mathcal{R}_q$ and $b \in \mathcal{R}_q$, the NTT multiplication is performed as follows.

$$\mathrm{NTT}^{-1}\Big(\mathrm{NTT}(a) \star \mathrm{NTT}(b)\Big) \tag{3}$$

To simplify the notation, we denote the NTT transformation for polynomials using 'hat' for the rest of the paper, *i.e.* $\hat{a} = \mathrm{NTT}(a)$. The element-wise multiplication in the NTT domain, $\hat{a} \star \hat{b}$, is known as the base multiplication.

Complete NTT. NTT is considered as an application of the Chinese Remainder Theorem (CRT) to $\mathcal{R}_q$. In case $q \equiv 1 \mod 2n$, a primitive $2n$-th root of unity $\zeta_{2n} \in \mathbb{Z}_q$ exists for which $\zeta_{2n}^n \equiv -1 \mod q$. This setting allows a *complete* NTT over $\mathcal{R}_q$, where $x^n + 1$ can be factored down to the linear factors, $\prod_{i=0}^{n-1}(x - \zeta_{2n}^{2i+1})$. The NTT transformation indeed computes the remainder from the division of its input polynomial by $(x - \zeta_{2n}^{2i+1})$ for each i, resulting in a vector of n elements, $\mathbb{Z}_q^n$. Consequently, the base multiplication is performed coefficient wise, a modular multiplication for each i.

Implementing NTT Transformation. The *forward* and *backward* NTT transformations can be efficiently implemented in $\log n$ steps. Each step is called an NTT *layer*. Indeed, the polynomial is recursively split until a linear degree is reached using so-called *butterfly units*. The forward transformation is usually implemented with *Cooley-Tuckey* (CT) butterflies [CT65] while the backward transformation is commonly realized using *Gentleman-Sande* (GS) butterflies [GS66] although it is not a must. For an input pair of coefficients a_0 and a_1, the CT butterfly computes the output pair by

$$\hat{a}_0 = a_0 - a_1 \cdot \delta, \qquad \hat{a}_1 = a_0 + a_1 \cdot \delta, \tag{4}$$

where δ is called the *twiddle factor*, a power of ζ_{2n}.

Incomplete NTT. Sometimes, due to performance optimizations or restrictions of the operated ring of polynomials, NTT is not computed for all $\log n$ layers [LS19, AHKS22, ABCG20, CHK+21, ACC+21]. This is referred to as *incomplete* NTT. In case the NTT is computed for $m < \log n$ layers, $\hat{a}$ for $a \in \mathcal{R}_q$ is a vector with 2^m elements and each element is a degree-$(\log n - m)$ polynomial. Then, the base multiplication refers to the multiplication of degree-$(\log n - m)$ polynomials. For instance, Kyber employs $q = 3329$ and $n = 256$ allowing a 7-layer NTT while $\log n = 8$. Therefore the base multiplication in this setting is achieved by performing 128 individual multiplications of degree-1 polynomials, as demonstrated in Algorithm 1.

Algorithm 1 Base Multiplication for $(\log n - 1)$-layer incomplete NTT

Input: $\hat{a}, \hat{b}$; the resulting vectors from $(\log n - 1)$-layer forward NTT transformation on $a, b \in \mathcal{R}_q$

Output: $\hat{r} = \hat{a} \star \hat{b}$

1: **for** $i \leftarrow 0$ until $n/2$ **do** ▷ Compute $\hat{r}_i = \hat{a}_i \cdot \hat{b}_i$

2: $\quad \hat{r}_{i,0} \leftarrow \hat{a}_{i,0} \cdot \hat{b}_{i,0} + \hat{a}_{i,1} \cdot \hat{b}_{i,1} \cdot \delta_i \mod q$ ▷ δ_i is a power of ζ_{2n}

3: $\quad \hat{r}_{i,1} \leftarrow \hat{a}_{i,1} \cdot \hat{b}_{i,0} + \hat{a}_{i,0} \cdot \hat{b}_{i,1} \cdot \mod q$

4: **end for**

4 Modular Arithmetic

In this section, we briefly review the modular reduction techniques that are adapted in LBC. Compared to ECC or RSA, the modular arithmetic in LBC deals with relatively shorter integers. Additionally, operating with signed integers in modular arithmetic proves to be more efficient in LBC [HZZ+22, AHKS22, GKS20, ABCG20, BKS19]. The main reason for this is due to the fact that it simply eliminates the need for an extra addition for preventing negativeness in the butterfly units (see Equation (4)). Table 2 summarizes state-of-the-reduction implementations based on the ARM Cortex-M4. It can be seen that the smallest latency in terms of the number of clock cycles is achieved by central reduction techniques [ABCG20, AHKS22, HZZ+22], whose output range is centered around 0. As a result, in addition to the NTT transformation, central reduction is also preferred to speed up the base multiplication.

Barrett Reduction. The Barrett reduction was originally proposed in [Bar87]. Its main idea is to subtract a factor of the modulus q from the number to reduce by approximating the division of the number by q through a pre-computed factor and shifting. A signed version of Barrett reduction adapted for LBC is proposed in [Sei18]. The input range of the signed Barrett reduction is $[-\beta/2, \beta/2)$, and the output range is $[0, q]$. A 9-cycle

Table 2: Summary of state-of-the-art reduction implementations on Cortex-M4.

Scheme	β	Input Range	Output Range	Packed	Cycles
Montgomery [ABCG20]	16	$[-q\cdot\beta/2, q\cdot\beta/2)$	$(-q, q)$	✗	2
Montgomery [GKS20]	32	$[-q\cdot\beta/2, q\cdot\beta/2)$	$(-q, q)$	✗	2
Montgomery [ABCG20]	16	$[-q\cdot\beta/2, q\cdot\beta/2)$	$(-q, q)$	✓	8
Barrett [Sei18]*	16	$[-\beta/2, \beta/2)$	$[0, q]$	✗	3
Barrett [AHKS22]	16	$[-\beta/2, \beta/2)$	$[-q/2, q/2]$	✓	6
Plantard [HZZ+22]	16	$[-q^2 2^{2\alpha'}, q^2 2^{2\alpha'}]$	$[-q/2, q/2]$	✗	2
Plantard [HZZ+22]	16	$[-q^2 2^{2\alpha'}, q^2 2^{2\alpha'}]$	$[-q/2, q/2]$	✓	5

* gives the definition of the algorithm but does not present an implementation on Cortex-M4.
α' is a parameter of Plantard reduction that satisfies $q < 2^{\beta-\alpha'-1}$.

implementation of the signed Barrett reduction for packed integers dedicated to ARM Cortex-M4 is presented by [ABCG20]. Packing of integers refers to storing two $\beta = 16$-bit integers in a $2\beta = 32$-bit register, which is then passed to the packed reduction function. Later, a 6-cycle implementation of Barrett reduction for packed integers was reported in [AHKS22], which performs a central reduction with an output range $[-q/2, q/2]$.

Montgomery Reduction. The Montgomery reduction is first proposed in [Mon85]. As the Barrett reduction, it enables a constant time reduction by eliminating the need for division. A signed version of Montgomery Reduction is presented by [Sei18], with an input range $[-q\cdot\beta/2, q\cdot\beta/2)$ and output range $(-q, q)$. While a 3-cycle implementation on ARM Cortex-M4 was initially given by [BKS19] for $\beta = 16$, the state-of-the-art implementation of Montgomery reduction [ABCG20, GKS20] takes 2 cycled for both $\beta = 16$ and $\beta = 32$. Also, an 8-cycle implementation of Montgomery reduction for packed integers was presented in [ABCG20].

Plantard Reduction. The Plantard reduction [Pla21] is a more recent algorithm compared to its counterparts, Montgomery and Barrett. While the original Plantard reduction operates with unsigned integers, the authors of [HZZ+22] proposed an improved version, which operates with signed integers to be employed in LBC. The output range of the signed version is $[-q/2, q/2]$, the same as the state-of-art Barrett reduction. One advantage of the Plantard reduction is that it enables 2-cycle modular multiplication by a constant, outperforming the 3-cycle Montgomery multiplication. The multiplication by a constant is beneficial for implementing the butterfly units during the NTT transformations (see Equation (4)). On the other hand, the improved Plantard reduction also takes 2 cycles on ARM Cortex-M4, the same as Montgomery. However, Plantard's 2-cycle implementation enables a larger input range and a smaller output range that is desirable. Specifically, the Plantard reduction outputs in the exact range $[-q/2, q/2]$. In other words, it does not require any final correction. This is a significant improvement over the 2-cycle Montgomery reduction whose output range is $(-q, q)$. As a side note, packed reduction takes 5 cycles.

5 Non-Profiled Side-Channel Attack on NTT Multiplication

In this section, we present the general outline of a non-profiled power/EM side-channel analysis attack on an implementation of a polynomial multiplication in the NTT domain.

Leakage Model. Let us first define the assumption for the leakage function of the target device based on a random variable X defined in a space $\mathcal{X}$, a constant scaling factor α, and a noise sampled from a Gaussian distribution with mean μ and standard deviation σ, which is independent of X as follows.

$$\mathcal{L}(X) = \alpha \cdot \mathcal{HW}_\beta(X) + \mathcal{N}(\mu, \sigma) \tag{5}$$

$\mathcal{L}(X)$ is commonly used to simulate side-channel leakage of micro-processors in presence of noise when X is processed.

Adversary Model. Consider the base multiplication $\hat{s} \star \hat{c}$ where $s \in \mathcal{R}_q$ is a secret polynomial, and $c \in \mathcal{R}_q$ is a public polynomial. The goal of an adversary is to reveal $\hat{s}$ through a non-profiled side-channel analysis attack where the attacker has access to the power/EM consumption pattern of the underlying device during the computation of $\hat{s} \star \hat{c}$. In a non-profiled attack, the attacker samples the leakage from $\hat{s} \star \hat{c}$ for N distinct computations where $\hat{c}$ changes for each measurement. The set of samples recorded for each measurement is referred to as a *trace*.

Attack Outline. As previously mentioned, the base multiplication in the NTT domain is performed element wise. Therefore, each $\hat{s}_i$ can be attacked independently using the knowledge of $\hat{c}_i$. To retrieve $\hat{s}_i$, a set of hypotheses is made, usually for the secret to reveal. Each hypothesis is tested by evaluating the *target function* thereby statistically comparing the observed leakages (traces). In a Correlation Power Analysis (CPA) attack [BCO04] , the output of the *target function* is transformed into *hypothetical power consumption* using a *hypothetical power model* with Hamming weight, $\mathcal{HW}_\beta(.)$, being the most frequently used hypothetical power model. The tools employed by side-channel analysis attacks for statistically comparing the hypothetical power consumption and observed leakages are called *distinguishers*. As the most commonly used distinguisher, CPA is based on estimating the correlation, e.g., by Pearson correlation coefficient, between measured and hypothetical leakages. In this study, the *target function* to reveal $\hat{s}_i$, is the output of multiplication $\hat{s}_i \cdot \hat{c}_i$. This can refer to a single modular multiplication modulo q (also seen as $\hat{s}_i$ which is a degree-0 polynomial) or a polynomial multiplication depending on whether the employed NTT is complete or incomplete. With this target function, the number of hypotheses also depends on the degree of $\hat{s}_i$ [MBB+22, TS23].

Masking. The most promising way to defeat the above-explained attack is masking [HKL+22, BGR+21a, FBR+21, ÖY23, MGTF19, ABC+23b, CGTZ23, RRd+16, BC22, CGMZ21, BDK+20, KDVB+22]. Indeed, masking of the polynomial multiplication is straightforward from an algorithmic perspective, as it can be seen as a linear operation. For a uniformly randomly generated share s^0, one computes the other share as $s^1 = s - s^0$. Then, the computation $\hat{s} \star \hat{c}$ is performed on the shares as $\hat{s}^0 \star \hat{c}$ and $\hat{s}^1 \star \hat{c}$. Notice that, $\hat{s} \star \hat{c} = \hat{s}^0 \star \hat{c} + \hat{s}^1 \star \hat{c}$. This type of masking is referred as *arithmetic masking*. In particular, the order or masking is defined by the number shares representing the secrets. Here in the given example, first-order masking is applied as two shares are used. Since $\hat{s}_i \cdot \hat{c}_i$ is not computed in the plain, the leakage of every single point in side-channel traces is expected to be independent of $\hat{s}_i$ and hence independent of the the estimated hypothetical power consumption. It is noteworthy to mention that an assumed condition for such a claim is that each share s^0 and s^1 individually should follow a uniform distribution.

Second-order Attack. In order to conduct successful attacks on a first-order masked implementation, the leakage associated to two shares should be combined using a pre-processing function. Since the shares are processed individually (not simultaneously), their associated leakages appear in different time samples. Intuitively for the application studied in this work, the attacker combines the observed leakages associated to $\hat{s}_i^0 \cdot \hat{c}_i$ and $\hat{s}_i^1 \cdot \hat{c}_i$ while mean-free product is known as the most efficient pre-processing (combination) function [PRB10]. For two random variables $Y_0 \in \mathbb{R}$ and $Y_1 \in \mathbb{R}$, which correspond to the leakage associated to $\hat{s}_i^0 \cdot \hat{c}_i$ and $\hat{s}_i^1 \cdot \hat{c}_i$, respectively, the mean-free product is defined as

$$\mathcal{C}(Y_0, Y_1) = (Y_0 - E[Y_0]) \cdot (Y_1 - E[Y_1]). \tag{6}$$

Needless to say, $E[Y_0]$ and $E[Y_1]$ are approximated by the sample means $\overline{Y}_0$ and $\overline{Y}_1$ over the trace set. A CPA attack performed on a masked implementation by means of such a pre-processing function is referred to as *higher-order CPA (HOCPA)*. In this study, we use HOCPA with the mean-free product as the distinguisher with different hypothetical power models.

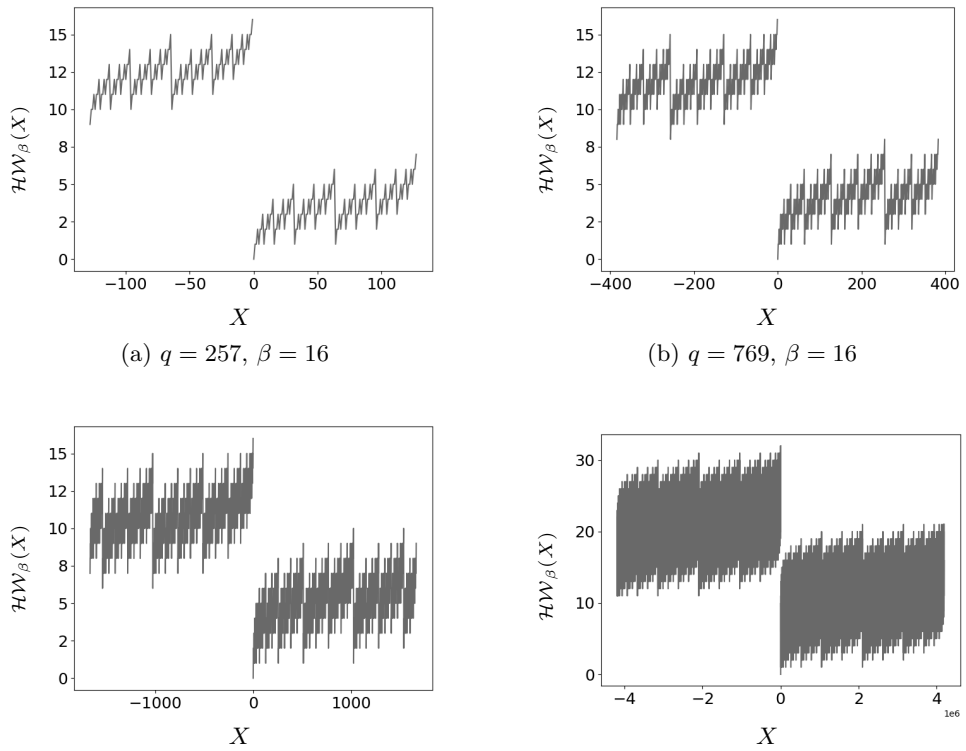
Application to Kyber and Dilithium. It is important to emphasize that the adversary model explained in this section as well as the target function directly applies to both Kyber and Dilithium. More precisely, performing an attack on the leakage of $\hat{s} \star \hat{c}$ to reveal $\hat{s}$ corresponds to targeting $\mathbf{s}^T \cdot u$ in the case of Kyber and targeting $c \cdot \mathbf{s}_1$ or $c \cdot \mathbf{s}_2$ for Dilithium (see Section 3).

6 Distribution of Hamming Weights for Signed Integers Modulo q

In this section, we study the distribution of Hamming weight of the signed representation of integers modulo q, *i.e.* the effect of central reduction on the Hamming weight. We evaluate the primes that are employed in Kyber and Dilithium with $q = 3329$ and $q = 8380417$, respectively. We also study the carrier primes that are employed for Dilithium to perform short polynomial arithmetic [AHKS22]. Namely, $q = 257$ for Dilithium2 and Dilithium5, and $q = 769$ for Dilithium3. We should note that masking the short polynomials in their range is possible [ABC+23a]. One important factor for computing the Hamming weight of negative integers is the machine word size β which does not have any effect on the Hamming weight of positive integers. For the rest of the paper, we take the machine word size $\beta = 16$ for $q = 257$, $q = 769$, and $q = 3329$ while $\beta = 32$ for $q = 8380417$.

Hamming Weight as an Indicator of the Sign. The main observation that led to this study is the clear separation of Hamming weight of the non-negative side of ${}^{\pm}\mathbb{Z}_q$, namely $[0, q/2]$ and the negative side $[-q/2, 0)$. As an intuition, consider $q = 257$. The positive interval of ${}^{\pm}\mathbb{Z}_{257}$ corresponds to $[0, 128]$, for which the maximum Hamming weight is $\mathcal{HW}_\beta(127) = 7$. In other words, the Hamming weight of integers $[0, 128]$ lies in $[0, 7]$. On the other hand, the negative side of ${}^{\pm}\mathbb{Z}_{257}$ corresponds to $[-128, 0)$, where the Hamming weights are in the range of $[9, 16]$ assuming 2's complement[1] representation with machine word size $\beta = 16$. Consequently, the Hamming weight of a number in ${}^{\pm}\mathbb{Z}_{257}$, reveals its sign immediately. Figure 1 visualizes our observation for all the primes analyzed in this study. Note that there is an overlap between the Hamming weight ranges $[-q/2, 0)$ and $[0, q/2]$, for $q = 769$, $q = 3329$, and $q = 8380417$. For instance, the Hamming weight of non-negative integers in ${}^{\pm}\mathbb{Z}_{3329}$ are distributed in $[0, 10]$ while the Hamming weight of

[1] Negative integers in 2's complement form are represented by inverting all bits of the corresponding positive integer and adding 1 to the result.



(a) $q = 257$, $\beta = 16$ (b) $q = 769$, $\beta = 16$

(c) $q = 3329$, $\beta = 16$ (d) $q = 8380417$, $\beta = 32$

Figure 1: Distribution of Hamming weights of integers in $[-q/2, q/2]$ in 2's complement representation.

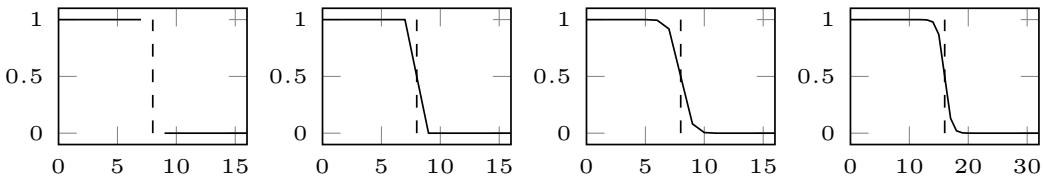
negative integers are in the range $[6, 16]$. Therefore, there is an overlap for five possible Hamming weights in the interval $[6, 10]$ out of a total of 17 possible values assuming the machine word size $\beta = 16$.

Despite this overlap between Hamming weights of non-negative and negative values for some primes, the Hamming weight can still be used to predict the sign of a number with a high probability. Therefore, we consider the Hamming weight as a noisy indicator of the sign when 2's complement is used for representation of negative integers modulo q. Figure 2 presents the probability of uniformly distributed random number $X \in {}^{\pm}\mathbb{Z}_q$ being non-negative given its Hamming weight. It is noteworthy to highlight that the uncertainty in the sign is either zero or very low except for $\mathcal{HW}_\beta(X) = \beta/2$. Therefore, we can define the following classifier to predict the sign of a number based on its Hamming weight.

$$\tilde{\mathcal{S}}(X) = \begin{cases} 0 & \text{if } \mathcal{HW}_\beta(X) > \beta/2 \\ 1, & \text{otherwise} \end{cases} \tag{7}$$

The accuracy of the above-presented sign predictor is given in Figure 3. While the lowest accuracy observed is 0.93 for $q = 3329$ and $\mathcal{HW}_\beta(X) \leq \beta/2$, it is still considered as a highly accurate prediction. For the other studied primes, the accuracy is even closer to 1.0.

Generalization for Arbitrary Primes. The accuracy of $\tilde{\mathcal{S}}(.)$ directly depends on the bit-lengths of q and β. The accuracy increases as the so-called gap $\mathcal{M}(q, \beta) = \beta + 1 - 2 \cdot (\mathcal{B}(q) - 1)$ increases. Particularly for $\mathcal{M}(q, \beta) > 0$ such as $q < 2^9$ and $\beta = 16$ (resp. $q < 2^{17}$ and



(a) $q = 257$ (b) $q = 769$ (c) $q = 3329$ (d) $q = 8380417$

Figure 2: $\mathcal{P}(X \geq 0)$ (in y-axis) with respect to $\mathcal{HW}_\beta(X)$ (in x-axis). The dashed line shows $\mathcal{HW}_\beta(X) = \beta/2$.

q	257	769	3329	8380417
$\mathcal{P}\big(X < 0 \mid \mathcal{HW}_\beta(X) > \beta/2\big)$	1.0	1.0	0.98	0.99
$\mathcal{P}\big(X \geq 0 \mid \mathcal{HW}_\beta(X) \leq \beta/2\big)$	1.0	0.99	0.93	0.97

Figure 3: Prediction accuracy of $\tilde{\mathcal{S}}(.)$, the sign classifier based on Hamming weight for different moduli q.

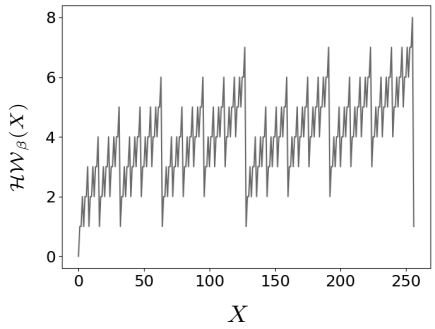


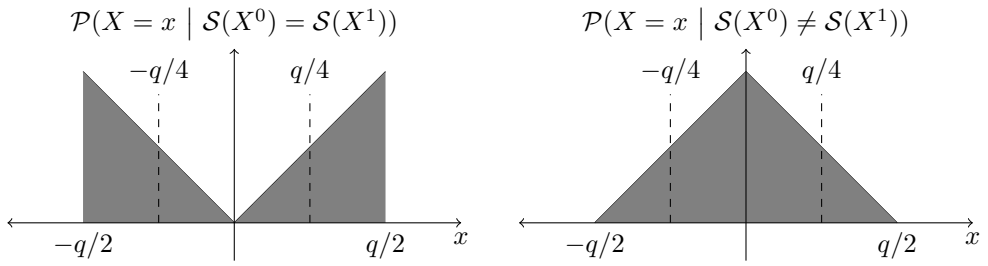
Figure 4: Distribution of Hamming weight of integers in $[0, q)$ for $q = 257$.

$\beta = 32$), the accuracy of $\tilde{\mathcal{S}}(.)$ is 1.0. This is because in these cases the Hamming weight distributions of $[-q/2, 0)$ and $[0, q/2]$ do not overlap. Among the lattice-based signatures submitted to the NIST extra call for the post-quantum signatures, EagleSign [HDS23] employs $q = 12289$. Considering $\beta = 16$ for this case, the accuracy for correctly predicting negatives and non-negatives using $\tilde{\mathcal{S}}(.)$ is 0.9 and 0.8, respectively. Another lattice-based signature Racoon [dPEK+] employs two primes, $q_1 = 16515073$ and $q_2 = 33292289$. For q_1 and $\beta = 32$, the accuracy is 0.98 and 0.95 for predicting negative and non-negative integers respectively. Similarly, for q_2 the accuracies are 0.96 and 0.93.

Distribution of Hamming Weight of Unsigned Integers Modulo q. We would like to note that the argument made in this section does not apply to the unsigned representation of integers modulo q, namely $\mathbb{Z}_q$. It can be seen in Figure 4 that no clear ranges can be identified for unsigned integers when observing their Hamming weight.

7 The Range Power Model

Based on our observation in the previous section, we present the range power model, which is very effective against arithmetic masking when central reduction is employed.



(a) Case 1:
$(X^0 < 0$ and $X^1 < 0)$ or $(X^0 \geq 0$ and $X^1 \geq 0)$
i.e. $\mathcal{S}(X^0) = \mathcal{S}(X^1)$

(b) Case 2:
$(X^0 \geq 0$ and $X^1 < 0)$ or $(X^0 \geq 0$ and $X^1 < 0)$
i.e. $\mathcal{S}(X^0) \neq \mathcal{S}(X^1)$

Figure 5: Probability distributions of $X = X^0 + X^1 \bmod^{\pm} q$ for $X^0, X^1 \in^{\pm} \mathbb{Z}_q$

Revealing the Secret Knowing the Sign of Shares. Consider two uniformly random variables $X^0, X^1 \in^{\pm} \mathbb{Z}_q$ and their modular addition $X^0 + X^1 \bmod^{\pm} q$. Given the signs of both variables, Figure 5 demonstrates the probability distribution of $X^0 + X^1 \bmod^{\pm} q$. As depicted in the figure, there are two cases for the distribution given the sign of both random variables. If the sign of X^0 and X^1 are the same, namely $\mathcal{S}(X^0) = \mathcal{S}(X^1)$, then the probability is distributed around $\pm q/2$. Otherwise, it is centered around 0. Indeed, the distributions correspond to the convolution of probability distribution functions. More precisely, one of

$$\mathcal{P}(X^0 = x^0 \mid X^0 < 0), \quad \mathcal{P}(X^0 = x^0 \mid X^0 \geq 0)$$

is convoluted to one of

$$\mathcal{P}(X^1 = x^1 \mid X^1 < 0), \quad \mathcal{P}(X^1 = x^1 \mid X^1 \geq 0).$$

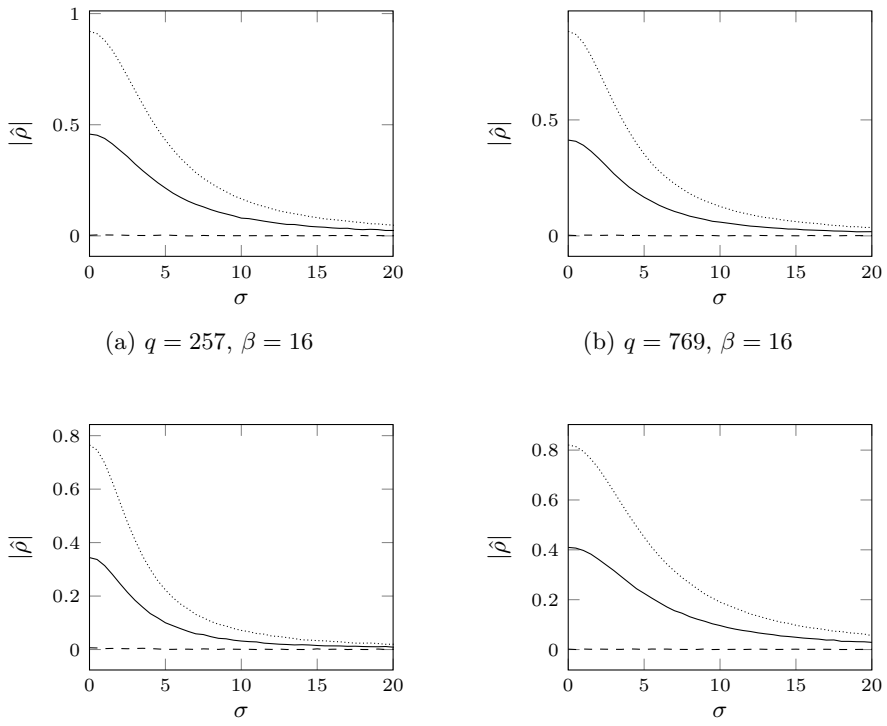
Now suppose that X^0 and X^1 are arithmetic shares representing a secret intermediate variable $X = (X^0, X^1)$. Then, the above discussion shows that information about the sign of the individual shares leads to a strong effect on the distribution of the secret X.

Estimating the Sign Equality from Side-Channel Leakages. In the previous section, we showed that the Hamming weight of 2's complement representation of an integer in $[-q/2, q/2]$ is a noisy indicator of its sign. Also, recall the leakage in CMOS circuits which is highly relevant to the Hamming weight of processed data (see Equation (5)). Now, let $Y_1 = \mathcal{L}(X^1)$ and $Y_1 = \mathcal{L}(X^1)$ denote the leakage associated to the random shares X^0 and X^1. To distinguish whether the sign of X^0 and X^1 are equal or not based on the leakage, the mean-free product $\mathcal{C}(Y_0, Y_1)$ can be used as follows.

$$\begin{aligned}\mathcal{C}(Y_0, Y_1) = &\Big(\alpha_0 \cdot \mathcal{HW}_\beta(X^0) + \mathcal{N}(\mu_0, \sigma_0) - \mathrm{E}\big[\alpha_0 \cdot \mathcal{HW}_\beta(X^0) + \mathcal{N}(\mu_0, \sigma_0)\big]\Big) \cdot \\ &\Big(\alpha_1 \cdot \mathcal{HW}_\beta(X^1) + \mathcal{N}(\mu_1, \sigma_1) - \mathrm{E}\big[\alpha_1 \cdot \mathcal{HW}_\beta(X^1) + \mathcal{N}(\mu_1, \sigma_1)\big]\Big) \end{aligned} \tag{8}$$

For the sake of simplicity, we assume $\alpha = \alpha_0 = \alpha_1$, $\mu = \mu_0 = \mu_1$, and $\sigma = \sigma_0 = \sigma_1$. As X^0 and X^1 are uniformly random signed integers represented by β bits in the computer memory, $E[X^0] = E[X^1] = \beta/2$. Then, we can write

$$\begin{aligned}\mathcal{C}(Y_0, Y_1) = &\Big(\alpha \cdot \mathcal{HW}_\beta(X^0) + \mathcal{N}(\mu, \sigma) - (\alpha \cdot \beta/2 + \mu)\Big) \cdot \\ &\Big(\alpha \cdot \mathcal{HW}_\beta(X^1) + \mathcal{N}(\mu, \sigma) - (\alpha \cdot \beta/2 + \mu)\Big), \end{aligned} \tag{9}$$



(a) $q = 257$, $\beta = 16$ (b) $q = 769$, $\beta = 16$

(c) $q = 3329$, $\beta = 16$ (d) $q = 8380417$, $\beta = 32$

Figure 6: Correlation between $\mathcal{C}(Y_0, Y_1)$ and different functions for $X = X^0 + X^1 \bmod^{\pm} q$ and with respect to noise standatd deviation σ. Estimations are performed with 1 million samples uniformly taken for X^0 and X^1. (dotted) with $\mathcal{I}(X^0, X^1)$, (solid) with $\mathcal{RN}_q(X)$, and (dashed) with $\mathcal{HW}_\beta(X)$.

and by distributing the terms and as $\mathcal{N}(0, \sigma) = \alpha \cdot \mathcal{N}(0, \sigma/\alpha)$,

$$\mathcal{C}(Y_0, Y_1) = \alpha^2 \Big(\mathcal{HW}_\beta(X^0) - \beta/2 + \mathcal{N}(0, \sigma/\alpha)\Big) \cdot \Big(\mathcal{HW}_\beta(X^1) - \beta/2 + \mathcal{N}(0, \sigma/\alpha)\Big). \quad (10)$$

It is easy to see that $\mathcal{C}(Y_0, Y_1)$ is expected to be positive if both $\mathcal{HW}_\beta(X^0)$ and $\mathcal{HW}_\beta(X^1)$ are larger than $\beta/2$ or both are smaller than $\beta/2$. Similarly, $\mathcal{C}(Y_0, Y_1)$ is expected to be negative when only one of $\mathcal{HW}_\beta(X^0)$ and $\mathcal{HW}_\beta(X^1)$ is less than $\beta/2$. For both cases, $\big|\mathcal{C}(Y_0, Y_1)\big|$ is expected to increase as $\big|\mathcal{HW}_\beta(X^0) - \beta/2\big| \cdot \big|\mathcal{HW}_\beta(X^1) - \beta/2\big|$ increases. Based on the sign predictor $\tilde{\mathcal{S}}(.)$ presented in Equation (7), the positive values of this function mostly correspond to the case where $\mathcal{S}(X^0) = \mathcal{S}(X^1)$, while the negative values mostly correspond to $\mathcal{S}(X^0) \neq \mathcal{S}(X^1)$, depending on σ and the modulus q. Figure 6 presents the estimated correlation between $\mathcal{C}(Y_0, Y_1)$ and the outcome of equality check between $\mathcal{S}(X^0)$ and $\mathcal{S}(X^1)$, namely $\mathcal{I}(X^0, X^1)$, based on Equation (10). For simplicity, we take $\alpha = 1$ as it does not affect the analysis. We can observe that $\mathcal{I}(X^0, X^1)$ strongly correlates to $\mathcal{C}(Y_0, Y_1)$. Even for larger σ, the same observation holds for all studied values of q employed in LBC. However, the magnitude of correlation slightly differs based on $\mathcal{M}(q, \beta)$.

Estimating the Sign Equality. As explained in Section 7 and shown by Figure 5, we conclude that

$$\mathcal{P}\big(|X| > q/4 \;\big|\; \mathcal{S}(X^0) = \mathcal{S}(X^1)\big) = 0.75, \quad \mathcal{P}\big(|X| < q/4 \;\big|\; \mathcal{S}(X^0) \neq \mathcal{S}(X^1)\big) = 0.75.$$

Therefore, based on the dependency of X on $\mathcal{S}(X^0)$ and $\mathcal{S}(X^1)$, we define a border at $q/4$ and compare it to X in order to predict $\mathcal{I}(X^0, X^1)$. If $|X| > q/4$, then $\mathcal{S}(X^0) = \mathcal{S}(X^1)$ with a probability of 0.75.

$$\mathcal{P}\Big(\mathcal{S}(X^0) = \mathcal{S}(X^1) \;\Big|\; |X| > q/4\Big) = \frac{\mathcal{P}\Big(|X| > q/4 \;\Big|\; \mathcal{S}(X^0) = \mathcal{S}(X^1)\Big) \cdot \mathcal{P}\Big(\mathcal{S}(X^0) = \mathcal{S}(X^1)\Big)}{\mathcal{P}\Big(|X| > q/4\Big)}$$
$$= \frac{0.75 \cdot 0.5}{0.5} = 0.75 \tag{11}$$

Similarly $\mathcal{S}(X^0) \neq \mathcal{S}(X^1)$ with a probability of 0.75, if $|X| < q/4$.

$$\mathcal{P}\Big(\mathcal{S}(X^0) \neq \mathcal{S}(X^1) \;\Big|\; |X| < q/4\Big) = \frac{\mathcal{P}\Big(|X| < q/4 \;\Big|\; \mathcal{S}(X^0) = \mathcal{S}(X^1)\Big) \cdot \mathcal{P}\Big(\mathcal{S}(X^0) = \mathcal{S}(X^1)\Big)}{\mathcal{P}\Big(|X| < q/4\Big)}$$
$$= \frac{0.75 \cdot 0.5}{0.5} = 0.75 \tag{12}$$

Hence, we introduce the range power model for $X \in^{\pm} \mathbb{Z}_q$ as follows.

$$\mathcal{RN}_q(X) = \begin{cases} 1, & \text{if } |X| > q/4 \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

Discussion on Setting the Border. We should highlight that $\mathcal{RN}_q(.)$ predicts the outcome of the equality check between $\mathcal{S}(X^0)$ and $\mathcal{S}(X^1)$ with 0.75 accuracy, based on the comparison $|X| > q/4$. If the border is set differently, then one of the cases is predicted with higher accuracy while for the other case the accuracy decreases. Assume that another border $b > q/4$ is used. Then, one can tell $\mathcal{S}(X^0) = \mathcal{S}(X^1)$ with more certainty (greater than 75%) in case $|X| > b$. On the other hand, it is less likely (less than 75%) to have $\mathcal{S}(X^0) \neq \mathcal{S}(X^1)$ for the case $|X| \leq b$. These observations can be made through Figure 5.

Comparison to $\mathcal{HW}_\beta(.)$. Figure 6 benchmarks our proposed power model $\mathcal{RN}_q(.)$ in terms of the correlation with $\mathcal{C}(Y_0, Y_1)$ compared to the classical approach $\mathcal{HW}_\beta(.)$ as well. It is shown that $\mathcal{RN}_q(.)$ outperforms $\mathcal{HW}_\beta(.)$ for the exemplary values of q. We would like to stress that the superiority is preserved as noise standard deviation σ increases.

8 Attack Simulation

In this section, we evaluate the efficiency of our approach through simulations. In particular, we compare the efficiency of our introduced range power model $\mathcal{RN}_q$ to the Hamming weight power model $\mathcal{HW}_\beta$, in a second-order attack explained in Section 5. In our simulations, we consider different noise levels and reduction scenarios for each of the studied primes.

Simulated traces. We perform the following routine to generate simulated traces for base multiplication:

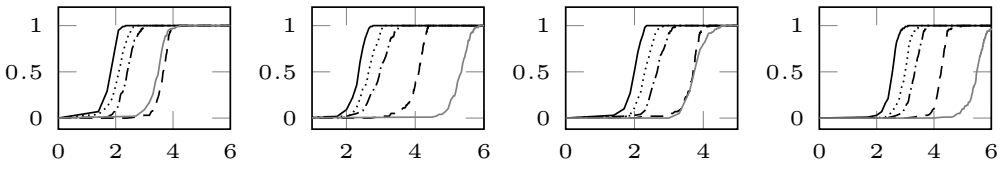
1. Generate a secret key vector $\mathbf{s} \in^{\pm} \mathbb{Z}_q^m$ uniformly at random.
2. Generate a public vector $\mathbf{c} \in^{\pm} \mathbb{Z}_q^m$ uniformly at random.
3. Sample $\mathbf{s}^0 \in^{\pm} \mathbb{Z}_q^m$ uniformly at random. Apply first-order masking to $\mathbf{s}$ such that $\mathbf{s} = \mathbf{s}^0 + \mathbf{s}^1 \bmod^{\pm} q$.

4. Compute $\mathbf{s}^0 \star \mathbf{c}$ and $\mathbf{s}^1 \star \mathbf{c}$. For each $i \in \{0, \ldots, m-1\}$ and $j \in \{0, 1\}$, perform a modular multiplication with (or without) central reduction as $\mathbf{r}_i^j \equiv \mathbf{s}_i^j \cdot \mathbf{c}_i \bmod^{\pm} q$ (or mod q) with m being the number of elements in $\mathbf{s}$ and $\mathbf{c}$. Particularly, the modular reduction is performed in $[-q/2, q/2]$, $(-q, q)$ or $[0, q)$ to simulate the leakage for different reduction algorithm presented in Table 2.
5. Compute the Hamming weight of each $\mathbf{r}_i^j$ as the simulated leakage ($2\,m$ individual results).
6. Apply Gaussian noise to the simulated leakages as in Equation (5), with $\alpha = 1$, $\mu = 0$, and the given standard deviation σ.
7. Go back to Step 2 until N traces are generated.

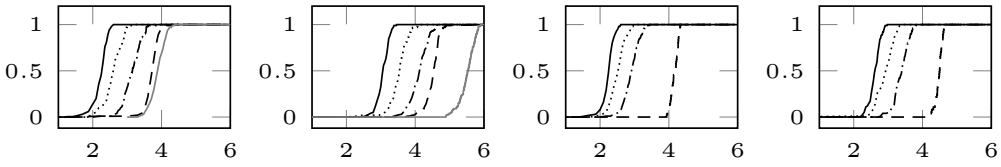
For computing the Hamming weights, 16-bit or 32-bit 2's complement representations are used depending on q, as explained in Section 6. m, N, σ, q as well as the reduction range (Step 4) are pre-defined parameters. We set $m = 100$ for all simulations. Note that the traces generated by this routine simulate a base multiplication for a *complete* NTT transformation, where the element-wise multiplication is just a modular multiplication. While $q = 8380417$ allows a complete NTT with the ring dimension $n = 256$, other moduli $q = 257$, $q = 769$, and $q = 3329$ do not allow complete NTT but allow incomplete NTT of 7 layers. Recall that the element-wise multiplication is the multiplication of degree-1 polynomials in that case. However, the simulations aim to benchmark our introduced range power model in comparison to the existing approaches. Therefore, there is no harm in doing the simulations as if the NTT is complete, which only affects the number of hypotheses. The comparison between the hypothetical power consumption and observed leakages is not affected by this behavior. Note also that we use $\mathbf{s}$ and $\mathbf{c}$ here instead of the notation $\hat{s}$ and $\hat{c}$ given in Section 5.

Evaluation. We apply HOCPA outlined in Section 5 on the simulated traces for both power models $\mathcal{RN}_q$ and $\mathcal{HW}_\beta$. Figure 7 presents the corresponding results, with respect to q, N, and σ, while the success rate refers to (# correctly predicted $\mathbf{s}_i/m$). As evident by the results, $\mathcal{RN}_q$ outperforms $\mathcal{HW}_\beta$ by more than one order of magnitude in case of central reduction to $[-q/2, q/2]$ (such as Plantard or Barrett, see Table 2). It should be noted that the number of traces is displayed on a logarithmic scale[2]. For instance, when $q = 257$ and $\sigma = 4$, the attack with $\mathcal{RN}_q$ needs around 500 traces to succeed while $\mathcal{HW}_\beta$ demands for at least 23500 traces, meaning 47× reduction in the number of required traces. We should highlight that $\mathcal{RN}_q$ remains the superior in different noise levels conforming with the observation shown in Figure 6. Also, the simulation results imply that $\mathcal{RN}_q$ performs better for all the studied primes. Further, the performance of $\mathcal{RN}_q$ depends on the accuracy of $\tilde{\mathcal{S}}(.)$, which is slightly worse for $q = 3329$ compared to the other evaluated primes. However, for $q = 3329$ and $\sigma = 4$, $\mathcal{RN}_q$ provides around 32× reduction in the number of required traces compared to $\mathcal{HW}_\beta$. In a perfect condition where $q = 257$ and $\sigma = 0$, $\mathcal{RN}_q$ only needs around 180 traces in the simulations achieving a 55× improvement. Similarly in a slightly less favorable case where $q = 769$ and $\sigma = 0$, around 230 traces are sufficient for $\mathcal{RN}_q$ to succeed which is around 45× less than what $\mathcal{HW}_\beta$ needs. The maximal improvement of $\mathcal{RN}_q$ over $\mathcal{HW}_\beta$ is observed with $q = 8380417$ and $\sigma = 4$, *i.e.* with around 67× reduction in N. On the other hand, $\mathcal{RN}_q$ is a better option for the reduction range $(-q, q)$ (such as Montgomery, see Table 2) albeit with a lower advantage compared to the other reduction algorithms. However, HOCPA with $\mathcal{RN}_q$ successfully retrieve the secret for all primes. Even if $\sigma = 4$, the attack only requires a few thousand traces and significantly outperforms $\mathcal{HW}_\beta$. For instance when $q = 3329$

[2]We should mention that for $q = 8380417$ and $\mathcal{HW}_\beta$ model when the reduction range is $[-q/2, q/2]$, we set $m = 35$ due to the long run time of simulations and attacks.



(a) $q = 257$, $\sigma = 0$ (b) $q = 257$, $\sigma = 4$ (c) $q = 769$, $\sigma = 0$ (d) $q = 769$, $\sigma = 4$



(e) $q = 3329$, $\sigma = 0$ (f) $q = 3329$, $\sigma = 4$ (g) $q = 8380417$, $\sigma = 0$ (h) $q = 8380417$, $\sigma = 4$

Figure 7: Success rate (in y-axis) of the second order attacks on simulated traces. x-axis denotes $\log_{10} N$. 100 experiments are performed with random data.
Hypothetical power models: $\mathcal{RN}_q$ (solid black, dotted), $\mathcal{HW}_\beta$ (dashed, dashdotted).
Reduction ranges: $[-q/2, q/2]$ (solid black, dashed), $(-q, q)$ (dotted, dashdotted), $[0, q)$ (solid gray).

and $\sigma = 4$, employing $\mathcal{RN}_q$ leads to around 3.7× less traces to succeed. Our simulation results also imply that the reduction interval $[-q/2, q/2]$ exhibits higher vulnerability to non-profiled attacks compared to $(-q, q)$. Note that $\mathcal{RN}_q$ performs better when the reduction's range is $[-q/2, q/2]$. Lastly, as a reference, we cover the scenario with the reduction to the unsigned range $[0, q)$ (such as Barrett from Table 2, with a negligible difference). We perform this set of experiments exclusively for the $\mathcal{HW}_\beta$ power model, since $\mathcal{RN}_q$ is explicitly constructed to attack the central reduction.

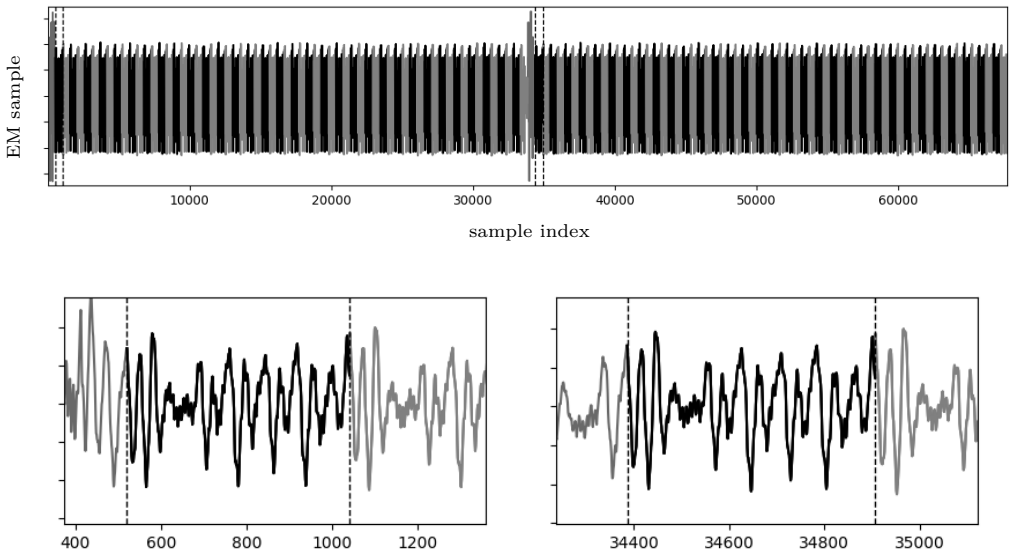
One important outcome of the conducted analyses is that the noise has a lower impact on the attacks on implementations with central reduction compared to the other reduction algorithms. Although the difficulty of attacks against both central and non-central reductions appears comparable for the $\mathcal{HW}_\beta$ power model and $\sigma = 0$, a notable difference is observed among the reduction algorithms for $\sigma = 4$ concerning the minimum number of required traces.

9 Practical Results: Application to Kyber

In this section, we present the result of applying our proposed approach to perform successful attacks on a protected implementation of Kyber.

Target Implementation. We focus on the ARM Cortex-M4 specific implementation of Kyber from the *pqm4* project [KRSS19a][3]. The implementation is mostly in assembly, and employs the Plantard reduction based on [HZZ+22] that we illustrated in Section 4. In particular, we focus on the function `frombytes_mul_asm_acc_32_16` which implements the base multiplication $\hat{s} \star \hat{c}$ in the incomplete NTT domain. First-order masking of $\hat{s} \star \hat{c}$ is implemented on top of pqm4 for experimental purposes. We should note that – to the best of our knowledge – all masked implementations of post-quantum algorithms

[3] commit hash: **3743a66**



(a) $\hat{s}_0^0 \cdot \hat{c}_0$ and $\hat{s}_1^0 \cdot \hat{c}_1$ (b) $\hat{s}_0^1 \cdot \hat{c}_0$ and $\hat{s}_1^1 \cdot \hat{c}_1$

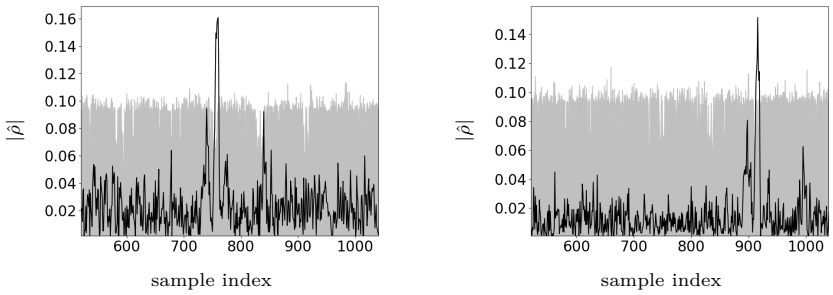
Figure 8: The mean EM trace associated to the execution of the base multiplication function `frombytes_mul_asm_acc_32_16` for both shares, $\hat{s}^0 \star \hat{c}$ and $\hat{s}^1 \star \hat{c}$. The iterations of the function are marked by interleaving black and gray colors. Due to loop unrolling, 64 iterations are observed for both shares instead of $n/2 = 128$. Recall that the ring dimension $n = 256$ for Kyber and 7-layer NTT is performed. The first iterations of `frombytes_mul_asm_acc_32_16` for both shares are marked and zoomed in (a) and (b).

on the ARM Cortex-M4 that have been reported in the literature are built on top of pqm4 by directly porting the linear operations including polynomial arithmetic [HKL+22, BGR+21a, ABC+23b, BDK+20, HDR23]. Therefore, we believe that assessing the most recent iteration of pqm4, featuring state-of-the-art polynomial arithmetic, would be beneficial. For instance, the open-source Kyber implementation [HKL+22] employs the Montgomery reduction since the more efficient Plantard reduction did not exist when the polynomial implementation was imported from pqm4. Nevertheless, we anticipate based on our simulation results that our attack is also effective against such an implementation with Montgomery reduction instead of Plantard. Our experiments are centered around the medium security level, *i.e.* Kyber768, though it does not affect our approach and results.

Setup. We used a LeCroy WavePro HD oscilloscope and a Langer ICR HH500-6 near-field micro-probe to collect electro-magnetic (EM) traces. The sampling rate was set to 1 GS/s. The victim program was running on a MAX32520 toolkit board from Analog Devices, which is equipped with an AMR Cortex-M4 operating at a frequency of 120 MHz. We provided a trigger signal for the oscilloscope to indicate the beginning of the function `frombytes_mul_asm_acc_32_16` for the first share. Hence, only the samples related to the base multiplication were recorded. The attacks have been performed using the scared library[4], with an in-house developed Python model that mimics the intended Kyber implementation.

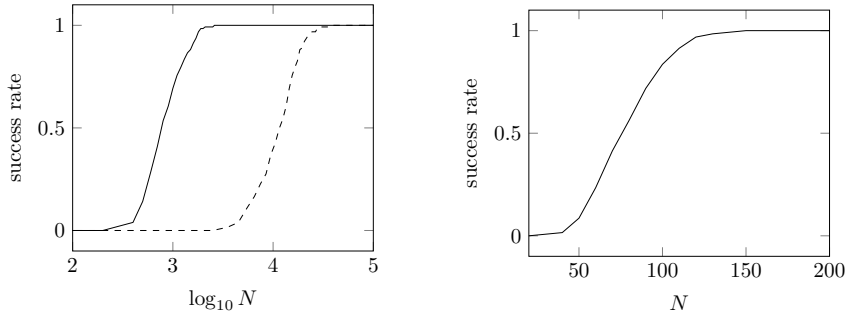
Attack Details. In order to perform the attacks, the raw traces were first aligned by detecting patterns centered around peaks. A mean trace over 1000 aligned traces is presented in

[4]https://pypi.org/project/scared/



(a) Attack on $\hat{s}_0$ (b) Attack on $\hat{s}_1$

Figure 9: HOCPA with $\mathcal{RN}_q$ against $\hat{s}_0$ and $\hat{s}_1$. The correlation scores of incorrect hypotheses are in gray, for the correct hypothesis is in black.



(a) HOCPA on the first-order masked implementation.
Power models: $\mathcal{RN}_q$ (solid), $\mathcal{HW}_\beta$ (dashed).
The attack succeeds with $\mathcal{RN}_q$ when $N = 2100$.

(b) CPA using $\mathcal{HW}_\beta$ model on an unmasked implementation.

Figure 10: The success rates of the discussed side-channel analysis attacks against Kyber. The success rate refers to (correctly predicted $\hat{s}_i/128$).

Figure 8. It should be noted that the iterations of the function `frombytes_mul_asm_acc_32_16` are visible through the mean trace for both shares. In order to reveal each $\hat{s}_i$, in the corresponding attacks we have only taken into account the relevant part of the EM traces based on the iterations. We used a constant offset to combine the leakages associated to two shares based on the pattern observed in Figure 8. It is noteworthy to mention that the same strategy can be easily adapted via educated guesses without prior knowledge of the specific implementation. Recall that $\hat{s}_i$ is a degree-1 polynomial in Kyber, and two coefficients must be predicted together based on the outline presented in Section 5. We tested $q \cdot q/2$ hypotheses ($\approx 2^{22.4}$ as $q = 3329$) so that either the actual secret or its additive inverse is found. The target of the attack is the lower-degree coefficient of each $\hat{s}_i \cdot \hat{c}_i$, precisely the $\hat{r}_{i,0}$ computed in Line 2 of Algorithm 1 for $\hat{a} = \hat{s}$ and $\hat{b} = \hat{c}$.

Evaluations. Let us start the evaluations by exemplary presenting the result of the individual attacks on $\hat{s}_0$ and $\hat{s}_1$ in Figure 9. The correlation peaks for the correct hypotheses are observed in the corresponding time samples for both secrets. Note that the correlations in absolute value for the correct hypotheses are around 0.155 in Figure 9, which corresponds to the simulation with $\sigma \approx 3.5$ in Figure 6c. From a higher perspective, Figure 10a compares our introduced power model $\mathcal{RN}_q$ to the existing approach $\mathcal{HW}_\beta$ in

terms of the number of traces needed to succeed. Consistent with the simulation results, $\mathcal{RN}_q$ is very effective against arithmetic masking with central reduction. We should point out that using $\mathcal{RN}_q$ results in only 2100 traces being needed to succeed, while $\mathcal{HW}_\beta$ requires about 30,000 traces. In other words, around **14**× reduction in the number of traces is achieved. Indeed, $N = 2100$ is considered as a very small number to conduct a successful non-profiling attack a masked implementation. The simulation results presented in Figure 7f imply that our approach requires roughly 2400 traces to succeed for $\sigma = 4$, which is in line with the experimental results presented here. Recall that we observed an order of magnitude reduction in the number of required traces using simulated traces as well.

Apart for the employed power model, finding the corresponding sample points to combine for a HOCPA is straightforward for the polynomial multiplication given the repeated pattern associated to `frombytes_mul_asm_acc_32_16` for each share. As a reference, we also included the success rate of a classical first-order CPA by the $\mathcal{HW}_\beta$ model performed on the same but unprotected implementation in Figure 10b. In order to keep the consistency, we used the same part of the EM traces as those considered in HOCPAs.

10 Conclusions and Future Work

In this paper, we investigated the impact of various reduction schemes on side-channel leakage, with a specific focus on the central reduction. State-of-the-art masking LBC, *e.g.* [MGTF19, HKL+22, BGR+21a, ABC+23b, BDK+20, HDR23, FBR+21, RRd+16, BC22, CGMZ21], concentrated on developing gadgets to handle non-linear operations and considered the linear portion of algorithms such as the polynomial arithmetic as relatively trivial to mask due to its straightforward duplication for the shares. However, our study reveals that the design decisions such as the reduction technique for the linear parts have a significant impact on the exploitability of the associated side-channel leakages and hence on the number of traces required for a successful attack. Our study exposes this fact by presenting a relatively easier second-order attack compared to classical and common second-order attack targeting the masked polynomial multiplication $\hat{s} \star \hat{c}$ in case of central reduction. While our attack particularly affects the masked implementations of lattice-based cryptography, it is generic to arithmetic masking with central reduction.

Our findings reveal that the signed representation of integers modulo q leads to a strong dependency between the sign of an integer and its Hamming weight in 2's complement form. We assessed this correlation through simulations involving the parameter sets employed by the post-quantum cryptography winners Kyber and Dilithium. We also efficiently exploited this source of leakage, by introducing a novel hypothetical power model, namely the range power model. We believe that our work is unique in the literature as it is the only 'non-profiled' attack particularly designed and efficient to exploit second-order leakages. We further have showcased our approach against a first-order masked implementation of Kyber. We have shown that our approach reduces the number of traces required for non-profiled side-channel analysis attacks to succeed by an order of magnitude compared to common and classical hypothetical power models. As our attack does not require profiling and is successful with only 2100 traces (in our experiments and using our measurement setup), we claim that utilization of the central reduction in masked implementations indeed increases side-channel vulnerability. To the best of our knowledge, we report the lowest number of traces for a successful non-profiled second-order attack against LBC.

As another outcome of our study, it demonstrates that finding the sample points in power/EM traces associated with the random arithmetic shares is trivial in masked implementations of LBC. Consequently, additional countermeasures such as shuffling must be applied. Indeed, it is relatively easier to shuffle the base multiplication since $\hat{s}_i \cdot \hat{c}_i$ are

performed independently for each i leading to $\frac{n}{2}!$ permutations.

We leave the generalization of our introduced power model to higher orders as a work for the future. Lastly, our study raises the question whether it is possible to develop algorithm-specific and more efficient solutions to conduct attacks on other applications of arithmetic masking?

References

[AAT+21] Furkan Aydin, Aydin Aysu, Mohit Tiwari, Andreas Gerstlauer, and Michael Orshansky. Horizontal side-channel vulnerabilities of post-quantum key exchange and encapsulation protocols. *ACM Transactions on Embedded Computing Systems (TECS)*, 20(6):1–22, 2021.

[ABC+23a] Aikata Aikata, Andrea Basso, Gaetan Cassiers, Ahmet Can Mert, and Sujoy Sinha Roy. Kavach: Lightweight masking techniques for polynomial arithmetic in lattice-based cryptography. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 366–390, 2023.

[ABC+23b] 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.

[ABCG20] Erdem Alkim, Yusuf Alper Bilgin, Murat Cenk, and François Gérard. Cortex-m4 optimizations for {R, M} lwe schemes. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 336–357, 2020.

[ACC+21] 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. *Cryptology ePrint Archive*, 2021.

[AHKS22] Amin Abdulrahman, Vincent Hwang, Matthias J. Kannwischer, and Amber Sprenkels. Faster kyber and dilithium on the cortex-M4. In Giuseppe Ateniese and Daniele Venturi, editors, *ACNS 22*, volume 13269 of *LNCS*, pages 853–871. Springer, Heidelberg, June 2022.

[BAA+19] Nina Bindel, Sedat Akleylek, Erdem Alkim, Paulo S. L. M. Barreto, Johannes Buchmann, Edward Eaton, Gus Gutoski, Juliane Kramer, Patrick Longa, Harun Polat, Jefferson E. Ricardini, and Gustavo Zanon. qTESLA. Technical report, National Institute of Standards and Technology, 2019. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-2-submissions.

[BAE+23] Olivier Bronchain, Melissa Azouaoui, Mohamed ElGhamrawy, Joost Renes, and Tobias Schneider. Exploiting small-norm polynomial multiplication with physical attacks: Application to crystals-dilithium. *Cryptology ePrint Archive*, 2023.

[Bar87] Paul Barrett. Implementing the Rivest Shamir and Adleman public key encryption algorithm on a standard digital signal processor. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 311–323. Springer, Heidelberg, August 1987.

[BBC+20] Daniel J. Bernstein, Billy Bob Brumley, Ming-Shing Chen, Chitchanok Chuengsatiansup, Tanja Lange, Adrian Marotzke, Bo-Yuan Peng, Nicola Tuveri, Christine van Vredendaal, and Bo-Yin Yang. NTRU Prime. 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.

[BC22] Olivier Bronchain and Gaëtan Cassiers. Bitslicing arithmetic/boolean masking conversions for fun and profit with application to lattice-based KEMs. *IACR TCHES*, 2022(4):553–588, 2022.

[BCO04] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In Marc Joye and Jean-Jacques Quisquater, editors, *CHES 2004*, volume 3156 of *LNCS*, pages 16–29. Springer, Heidelberg, August 2004.

[BDK+20] Michiel Van Beirendonck, Jan-Pieter D'Anvers, Angshuman Karmakar, Josep Balasch, and Ingrid Verbauwhede. A side-channel resistant implementation of SABER. Cryptology ePrint Archive, Report 2020/733, 2020. https://eprint.iacr.org/2020/733.

[BGR+21a] 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. https://tches.iacr.org/index.php/TCHES/article/view/9064.

[BGR+21b] Joppe W. Bos, Marc Gourjon, Joost Renes, Tobias Schneider, and Christine van Vredendaal. Masking kyber: First- and higher-order implementations. Cryptology ePrint Archive, Report 2021/483, 2021. https://eprint.iacr.org/2021/483.

[BKS19] Leon Botros, Matthias J. Kannwischer, and Peter Schwabe. Memory-efficient high-speed implementation of Kyber on Cortex-M4. Cryptology ePrint Archive, Report 2019/489, 2019. https://eprint.iacr.org/2019/489.

[BNGD22] Linus Backlund, Kalle Ngo, Joel Gärtner, and Elena Dubrova. Secret key recovery attacks on masked and shuffled implementations of CRYSTALS-kyber and saber. Cryptology ePrint Archive, Report 2022/1692, 2022. https://eprint.iacr.org/2022/1692.

[CDH+20] Cong Chen, Oussama Danba, Jeffrey Hoffstein, Andreas Hulsing, Joost Rijneveld, John M. Schanck, Peter Schwabe, William Whyte, Zhenfei Zhang, Tsunekazu Saito, Takashi Yamakawa, and Keita Xagawa. NTRU. 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.

[CGMZ21] Jean-Sébastien Coron, François Gérard, Simon Montoya, and Rina Zeitoun. High-order polynomial comparison and masking lattice-based encryption. Cryptology ePrint Archive, Report 2021/1615, 2021. https://eprint.iacr.org/2021/1615.

[CGTZ23] Jean-Sébastien Coron, François Gérard, Matthias Trannoy, and Rina Zeitoun. Improved gadgets for the high-order masking of dilithium. *IACR TCHES*, 2023(4):110–145, 2023.

[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: New speed records for saber and ntru on cortex-m4 and avx2. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 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 *2021 IEEE 39th International Conference on Computer Design (ICCD)*, pages 583–590. IEEE, 2021.

[CT65] James W Cooley and John W Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, 19(90):297–301, 1965.

[DNG22] Elena Dubrova, Kalle Ngo, and Joel Gärtner. Breaking a fifth-order masked implementation of CRYSTALS-kyber by copy-paste. Cryptology ePrint Archive, Report 2022/1713, 2022. https://eprint.iacr.org/2022/1713.

[dPEK+] Rafael del Pino, Thomas Espitau, Shuichi Katsumata, Mary Maller, Fabrice Mouhartem, Thomas Prest, Mélissa Rossi, and Markku-Juhani Saarinen. A side-channel secure signature scheme.

[FBR+21] Tim Fritzmann, Michiel Van Beirendonck, Debapriya Basu Roy, Patrick Karl, Thomas Schamberger, Ingrid Verbauwhede, and Georg Sigl. Masked accelerators and instruction set extensions for post-quantum cryptography. Cryptology ePrint Archive, Report 2021/479, 2021. https://eprint.iacr.org/2021/479.

[FBR+22] Tim Fritzmann, Michiel Van Beirendonck, Debapriya Basu Roy, Patrick Karl, Thomas Schamberger, Ingrid Verbauwhede, and Georg Sigl. Masked accelerators and instruction set extensions for post-quantum cryptography. *IACR TCHES*, 2022(1):414–460, 2022.

[GKS20] Denisa O. C. Greconici, Matthias J. Kannwischer, and Amber Sprenkels. Compact dilithium implementations on cortex-M3 and cortex-M4. Cryptology ePrint Archive, Report 2020/1278, 2020. https://eprint.iacr.org/2020/1278.

[GS66] W Morven Gentleman and Gordon Sande. Fast fourier transforms: for fun and profit. In *Proceedings of the November 7-10, 1966, fall joint computer conference*, pages 563–578, 1966.

[HDR23] Daniel Heinz and Gabi Dreo Rodosek. Fast first-order masked nttru. In *International Workshop on Constructive Side-Channel Analysis and Secure Design*, pages 127–148. Springer, 2023.

[HDS23] Abiodoun Clement Hounkpevi, Sidoine Djimnaibeye, and Michel Seck. Eaglesign: A new post-quantum elgamal-like signature over lattices. *Submission to the NIST's post-quantum cryptography standardization process*, 2023.

[HKL+22] Daniel Heinz, Matthias J. Kannwischer, Georg Land, Thomas Pöppelmann, Peter Schwabe, and Amber Sprenkels. First-order masked kyber on ARM cortex-M4. Cryptology ePrint Archive, Report 2022/058, 2022. https://eprint.iacr.org/2022/058.

[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.

[KAA22] Emre Karabulut, Erdem Alkim, and Aydin Aysu. Single-trace side-channel attacks on ω-small polynomial sampling: With applications to NTRU, NTRU prime, and CRYSTALS-DILITHIUM. Cryptology ePrint Archive, Report 2022/494, 2022. https://eprint.iacr.org/2022/494.

[KDVB+22] Suparna Kundu, Jan-Pieter D'Anvers, Michiel Van Beirendonck, Angshuman Karmakar, and Ingrid Verbauwhede. Higher-order masked saber. In *International Conference on Security and Cryptography for Networks*, pages 93–116. Springer, 2022.

[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, Heidelberg, August 1999.

[KLH+20] Il-Ju Kim, Tae-Ho Lee, Jaeseung Han, Bo-Yeon Sim, and Dong-Guk Han. Novel single-trace ML profiling attacks on NIST 3 round candidate dilithium. Cryptology ePrint Archive, Report 2020/1383, 2020. https://eprint.iacr.org/2020/1383.

[KRSS19a] 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. https://eprint.iacr.org/2019/844.

[KRSS19b] Matthias J Kannwischer, Joost Rijneveld, Peter Schwabe, and Ko Stoffelen. pqm4: Testing and benchmarking nist pqc on arm cortex-m4. 2019.

[LDK+22] Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Peter Schwabe, Gregor Seiler, Damien Stehlé, and Shi Bai. CRYSTALS-DILITHIUM. Technical report, National Institute of Standards and Technology, 2022. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022.

[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, Heidelberg, May / June 2010.

[LS19] Vadim Lyubashevsky and Gregor Seiler. Nttru: truly fast ntru using ntt. *Cryptology ePrint Archive*, 2019.

[Lyu09] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. Springer, Heidelberg, December 2009.

[MBB+22] Catinca Mujdei, Arthur Beckers, Jose Bermundo, Angshuman Karmakar, Lennert Wouters, and Ingrid Verbauwhede. Side-channel analysis of lattice-based post-quantum cryptography: Exploiting polynomial multiplication. Cryptology ePrint Archive, Report 2022/474, 2022. https://eprint.iacr.org/2022/474.

[MGTF19] Vincent Migliore, Benoît Gérard, Mehdi Tibouchi, and Pierre-Alain Fouque. Masking dilithium: Efficient implementation and side-channel evaluation. In *Applied Cryptography and Network Security: 17th International Conference, ACNS 2019, Bogota, Colombia, June 5–7, 2019, Proceedings 17*, pages 344–362. Springer, 2019.

[Mon85] Peter L Montgomery. Modular multiplication without trial division. *Mathematics of computation*, 44(170):519–521, 1985.

[MUTS22] Soundes Marzougui, Vincent Ulitzsch, Mehdi Tibouchi, and Jean-Pierre Seifert. Profiling side-channel attacks on dilithium: A small bit-fiddling leak breaks it all. *Cryptology ePrint Archive*, 2022.

[ÖY23] 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. IEEE, 2023.

[PFH+22] Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. FALCON. Technical report, National Institute of Standards and Technology, 2022. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022.

[Pla21] Thomas Plantard. Efficient word size modular arithmetic. *IEEE Transactions on Emerging Topics in Computing*, 9(3):1506–1518, 2021.

[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, Heidelberg, September 2017.

[PRB10] Emmanuel Prouff, Matthieu Rivain, and Régis Bévan. Statistical analysis of second order differential power analysis. Cryptology ePrint Archive, Report 2010/646, 2010. https://eprint.iacr.org/2010/646.

[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.

[RRd+16] Oscar Reparaz, Sujoy Sinha Roy, Ruan de Clercq, Frederik Vercauteren, and Ingrid Verbauwhede. Masking ring-LWE. *Journal of Cryptographic Engineering*, 6(2):139–153, June 2016.

[SAB+22] Peter Schwabe, Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, Damien Stehlé, and Jintai Ding. CRYSTALS-KYBER. Technical report, National Institute of Standards and Technology, 2022. available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022.

[Sei18] Gregor Seiler. Faster avx2 optimized ntt multiplication for ring-lwe lattice cryptography. *Cryptology ePrint Archive*, 2018.

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

[SLKG22] Hauke Steffen, Georg Land, Lucie Kogelheide, and Tim Güneysu. Breaking and protecting the crystal: Side-channel analysis of Dilithium in hardware. Cryptology ePrint Archive, Report 2022/1410, 2022. https://eprint.iacr.org/2022/1410.

[SLKG23] Hauke Steffen, Georg Land, Lucie Kogelheide, and Tim Güneysu. Breaking and protecting the crystal: Side-channel analysis of dilithium in hardware. In *International Conference on Post-Quantum Cryptography*, pages 688–711. Springer, 2023.

[TS23] Tolun Tosun and Erkay Savas. Zero-value filtering for accelerating non-profiled side-channel attack on incomplete ntt based implementations of lattice-based cryptography. *Cryptology ePrint Archive*, 2023.

[XPRO20] Zhuang Xu, Owen Pemberton, Sujoy Sinha Roy, and David Oswald. Magnifying side-channel leakage of lattice-based cryptosystems with chosen ciphertexts: The case study of kyber. Cryptology ePrint Archive, Report 2020/912, 2020. https://eprint.iacr.org/2020/912.