Non-profiled Single-Trace ML-DSA Secret Key Recovery by Horizontal Distribution Analysis

Linus Backlund and Elena Dubrova

KTH Royal Institute of Technology, Stockholm, Sweden, lbackl@kth.se, dubrova@kth.se

Abstract. During the last decade, single-trace side-channel attacks on cryptographic implementations have largely focused on profiled methods, which typically assume that the attacker has a clone device and implementation knowledge to build a leakage model for the target before launching the attack. In this paper, we demonstrate that a much weaker adversary, without the ability to profile, can also mount powerful single-trace attacks. The presented method compares the observed sample distribution with the expected Hamming weight leakage distribution using a Gaussian mixture model as a distinguisher. The horizontal nature of the attack allows us to obtain multiple observations per execution by partitioning and stacking the trace. We show how to select optimal parameters for trace preprocessing and configure the measurement setup so that the noise floor is lowered, enabling the exploitation of weak leakage. We also prove that imperfect classifications can be efficiently corrected using information-set decoding algorithms. The new method is evaluated on an optimized implementation of ML-DSA-65 on ARM Cortex-M4 using the secret key unpacking procedure as an attack point. With an enumeration budget of 2^{70} operations in $\mathbb{Z}_q$, the full secret key is recovered from a single trace in 99 % of attacks.

Keywords: Side-channel analysis · non-profiled attacks · horizontal attacks · post-quantum cryptography · Gaussian mixture model · ML-DSA · Dilithium · information-set decoding

1 Introduction

Recent works in side-channel analysis (SCA) have mainly considered profiled attack methods in which the attacker creates a model of the target's leakage ahead of the attack phase. A major driver of this interest has been the compelling efficiency of machine learning-based SCA approaches that enable single-trace attacks on cryptographic implementations [SKL+20, HLK+21, WNGD24, JD26].

Profiling methods, however, typically assume that the attacker has access to a clone device running an implementation identical to the target one and knows the details of the implementation. If the target device itself is used for profiling, then it is assumed that the attacker can control some intermediate values during the profiling phase. These assumptions constrain the scenarios in which profiled attacks can be mounted.

Non-profiled attacks can relax some of these requirements. However, previous non-profiled single-trace attacks [SKL+20, GR24, PC16, QGQ+25, AEVR23] usually rely either on strong side-channel leakage or on an algorithmic structure that enables soft-analytical postprocessing. In this paper, we demonstrate that a much weaker adversary can also mount powerful single-trace attacks.

We present a non-profiled horizontal attack method for a black-box setting, which assumes that the attacker has no access to a clone device and no detailed knowledge of the target implementation. The presented attack exploits leakage by comparing the observed sample distribution at each potential point of interest (POI) with the expected

Table 1: Summary of state-of-the-art single-trace power side-channel attacks against PQC implementations on the Cortex-M4 (without soft-analytical postprocessing).

Attack	Non-profiled	Distinguisher	Multiclass classification	Minimum ΔHW	Target Algorithm
[SKL+20]	No	Neural Network	No	1	SABER
[SKL+20]			**Yes**	1	FrodoKEM
[HLK+21]			**Yes**	1	ML-DSA
[WNGD24]			**Yes**	1	ML-DSA
[JD26]			No	-	MAYO
[JD26]			**Yes**	1	MAYO
[ACK+23]		Template	**Yes**	1	UOV
[GR24]			**Yes**	17	HAWK
[QA26]			No	64	FALCON
[PK25]			**Yes**	1	ML-KEM
[SKL+20]	**Yes**	k-means	No	16	ML-KEM
[GR24]		Threshold	No[a]	17	HAWK
This work		GMM	**Yes**	1	ML-DSA

[a]Adds ≈1% of coefficients classified with ternary k-means (not essential for the attack success).

Hamming weight (HW) leakage distribution. This provides a means of identifying POIs corresponding to the targeted intermediate variables. At the identified POIs, a Gaussian Mixture Model (GMM), fitted directly to the attack trace, serves as a non-profiled multiclass distinguisher. The traces are preprocessed using bandpass filtering to improve the distinguisher performance. We show that the choice of cutoff frequencies is important and propose an approach for selecting them without the knowledge of intermediate values. We also show how to configure the measurement setup so that leakage values with the Hamming weight difference of $\Delta\mathrm{HW} = 1$ can be distinguished. The $\Delta\mathrm{HW} = 1$ leakage is considerably weaker than the $\Delta\mathrm{HW} \geq 16$ leakage exploited by previous non-profiled single-trace attacks targeting multiclass classification [SKL+20, GR24].

The horizontal nature of the attack allows us to obtain multiple observations from a single execution by segmenting the trace and stacking the segments. This enables *direct* recovery of fixed secrets, such as the key, from a single trace. Classical differential power analysis (DPA) [KJJ99] and correlation power analysis (CPA) [BCO04] on the other hand, require multiple observations obtained under attacker-known variations in input values to recover a secret. The presented distribution-based distinguisher is simple and interpretable. Its construction may require no knowledge other than the algorithm specification.

We evaluate the new method on an optimized pqm4 implementation of ML-DSA [KPR+] (formerly CRYSTALS-Dilithium [DKL+18]), the first post-quantum digital signature scheme standardized by the National Institute of Standards and Technology (NIST) [Nat24a]. ML-DSA is a relevant case study, given its recent standardization and its expected role in post-quantum cryptography (PQC) deployments. While the pqm4 implementation [KPR+] does not employ countermeasures against side-channel attacks, the targeted secret key unpacking procedure is left unprotected in otherwise protected ML-DSA implementations, including [ABC+23, CGL+24].

Our main contributions are:

- **The first non-profiled single-trace multiclass attack.** As summarized in Table 1, no prior non-profiled single-trace attack can perform multiclass classification beyond a marginal extension of an otherwise binary distinguisher [GR24]. Two techniques contribute to the attack's success. First, we perform several minor modifications of a standard measurement setup that lower the noise floor. Second, we present a method for selecting bandpass filter parameters without the knowledge of intermediate values. These techniques reduce the number of classification errors,

allowing us to distinguish leakage values with $\Delta\text{HW} = 1$ from a single trace with a high probability.

- **A method for correcting imperfect classifications using information-set decoding (ISD) algorithms.** We prove that an overdetermined system of linear equations, whose coefficients potentially contain errors, can be reduced to a syndrome decoding problem. This enables the use of efficient ISD algorithms for correcting imperfect classifications. Although presented in the context of ML-DSA, the method may be applicable to other algorithms whose structural constraints can be expressed as a system of linear equations.

- **Single-trace key recovery attack on an ML-DSA-65 implementation.** With an enumeration budget of 2^{70} field operations in $\mathbb{Z}_q$, we recover the full secret key of ML-DSA-65 running on an ARM Cortex-M4 from a single trace in 99 % of attacks.

Table 1 places our work among state-of-the-art single-trace attacks on PQC algorithms.

1.1 Paper Organization

The rest of the paper is organized as follows. Section 2 presents the background. Section 3 reviews relevant previous work. Section 4 defines the threat model and attacker assumptions. Section 5 describes the noise reduction techniques used in this work. Section 6 presents the attack on ML-DSA-65. Section 7 describes the full secret key recovery process, including the new method for correcting classification errors. Section 8 evaluates the attack performance. Section 9 discusses possible countermeasures. Section 10 concludes the paper.

2 Background

This section describes the technical background necessary for understanding the paper.

2.1 Profiled vs. Non-profiled Attacks

Side-channel attacks can be classified as *profiled* or *non-profiled*, depending on how the leakage model used by the distinguisher is constructed.

In the profiled setting, the attacker builds a leakage model before launching the attack, requiring that they can know the values of an intermediate variable. This is typically achieved by using one or more fully controlled, target-identical clone devices [CRR03]. Alternatively, the algorithm itself may sometimes permit this capability in specific scenarios, such as when targeting decapsulation [JD25]. This allows the attacker to collect labeled traces for known intermediate values and to construct a model, such as a template [CRR03] or a neural network [MPP16], that characterizes the leakage associated with each class of interest. The learned model is then applied to traces collected from the device under attack to classify targeted intermediate values. Profiled attacks can achieve high classification accuracy because the leakage model is tailored to the target device and implementation. However, they require capabilities that may be difficult to obtain in practice, e.g. access to a clone device, the ability to acquire a sufficiently large profiling dataset, and control over the data processed during profiling.

In the non-profiled setting, the attacker instead constructs the distinguisher using traces obtained directly from the device under attack and a generic leakage model, e.g. the Hamming weight model described in Section 2.4 [SMY09]. Non-profiled attacks therefore usually assume a weaker adversary than profiled attacks. This typically comes at the cost of requiring stronger exploitable leakage, more measurements, or soft-analytical postprocessing to compensate for the absence of a device-specific leakage model. Achieving

a *non-profiled single-trace* attack, as in this work, is particularly challenging because neither profiling data, nor averaging across multiple executions, nor a structure permitting soft-analytical postprocessing is available to the attacker.

2.2 Horizontal vs. Vertical Attacks

Side-channel attacks can also be classified as *vertical* or *horizontal*, depending on whether the distinguisher combines observations across multiple executions or within a single execution.

Vertical attacks, including DPA [KJJ99] and CPA [BCO04], combine corresponding observations from *multiple* traces and evaluate their statistical dependence on a hypothesized intermediate value. Such an approach is effective when the targeted intermediate value varies across executions, i.e. because it depends on an attacker-controlled input. For the cases when the targeted intermediate value is fixed across executions, however, the statistical analysis provides little or no distinguishing information because the observations no longer correspond to varying hypotheses.

Horizontal attacks instead exploit multiple observations obtained within a *single* trace [PZS17]. They are applicable when an execution contains multiple operations whose leakage can be related to a common secret. Furthermore, by partitioning a trace into segments corresponding to repeated operations, the attacker can obtain multiple observations and construct a distinguisher from a single execution.

In this paper, the repeated processing of secret key coefficients of ML-DSA provides the structure required for a horizontal attack. We partition a single execution into the corresponding segments and use them as observations for a distinguisher. This allows the distinguisher to be constructed directly from a single attack trace.

2.3 Gaussian Mixture Models

A *Gaussian mixture model* represents a probability distribution as a weighted sum of Gaussian component distributions [Bis06]. In the context of side-channel analysis, each component can represent the leakage distribution associated with a particular class of intermediate values, e.g. their HW [LRP07]. Unlike binary distinguishers, which separate two hypotheses, a GMM can simultaneously model multiple classes and assign an observed sample to the most likely class. This enables multiclass classification of intermediate values with different HWs, which is central to the attack presented in this work.

For a K-class classification, given an observed sample x, its class can be estimated from the posterior probability

$$P(j \mid x) = \frac{w_j \mathcal{N}(x \mid \mu_j, \sigma_j^2)}{\sum_p w_p \mathcal{N}(x \mid \mu_p, \sigma_p^2)}. \tag{1}$$

where $j, p \in \{1, 2, \ldots, K\}$. The sample can then be assigned to the class with the highest posterior probability [Bis06].

In this paper, GMMs are used as a non-profiled classifier. Rather than learning the leakage distribution using traces acquired from a profiling device, as in template attacks [CRR03], the expected classes and their relative frequencies are derived from the algorithm specification. The resulting classification therefore requires no implementation-specific profiling data. This makes the GMM particularly suitable for the black-box setting considered in this work, where only the specification and traces from the target device are available.

2.4 Power Side-Channel Leakage

The power consumption of a CMOS integrated circuit typically depends on the data processed by its logic and memory elements. In side-channel analysis of software implementations, this data-dependent leakage is usually approximated using the Hamming weight model [BCO04, MOP07], which assumes that the leakage is proportional to the number of set bits in an intermediate value. A commonly used noisy HW leakage model is defined as

$$L(x) = a \cdot HW(x) + b + \epsilon, \tag{2}$$

where $HW(x)$ denotes the HW of x, a is the leakage gain, b is a constant offset and $\epsilon \sim \mathcal{N}(0, \sigma^2)$ is zero-mean Gaussian noise with variance σ^2.

Under this model, observed samples corresponding to values with the same HW follow the same Gaussian distribution. If an intermediate variable can take values with different HWs, the resulting distribution of samples is therefore the joint distribution of a GMM. Since the noise term in eq. (2) has a fixed variance σ^2, all HW classes share the same variance, while their means differ according to the leakage gain a. For K possible HW values of x, its probability density function (PDF) is given by

$$f(x) = \sum_{j=0}^{K-1} w_j \mathcal{N}(x|\mu_j, \sigma^2), \tag{3}$$

where w_j is the relative frequency of the HW j, μ_j is the class mean, and σ^2 is the variance of the class's distribution. Under the HW leakage model (2), the class means can be approximated as $\mu_j = aj + b$.

2.5 ML-DSA Algorithm

ML-DSA (previously Dilithium) is a post-quantum digital signature algorithm standardized by NIST in FIPS-204 [Nat24a]. It is a module-lattice based scheme defined for three parameter sets, ML-DSA-44, ML-DSA-65 and ML-DSA-87. The parameters relevant to this work are listed in Table 2.

2.5.1 Secret and Public Keys

ML-DSA operates over the polynomial ring $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ where $n = 256$ and $q = 8\,380\,417$. The secret key sk contains the public random seed ρ, the private random seed K, the hash of the public key tr, the secret polynomial vectors $\mathbf{s}_1 \in R_q^\ell$ and $\mathbf{s}_2 \in R_q^k$, and the d lower-order bits $\mathbf{t}_0$ of the public polynomial vector $\mathbf{t} \in R_q^k$. The coefficients of $\mathbf{s}_1$ and $\mathbf{s}_2$ are independently and uniformly sampled from the set $\{-\eta, \ldots, \eta\}$. Thus, they have a small, known range: $\{-2, -1, 0, 1, 2\}$ for ML-DSA-44 and ML-DSA-87, and $\{-4, \ldots, 4\}$ for ML-DSA-65.

The public key pk contains ρ and the higher-order bits $\mathbf{t}_1$ of $\mathbf{t}$, where

$$\mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2 \tag{4}$$

and matrix $\mathbf{A} \in R_q^{k \times \ell}$ is deterministically sampled from ρ.

Although $\mathbf{t}_0$ is stored as a part of the secret key sk, it is not considered a secret. FIPS 204 [Nat24a] notes that $\mathbf{t}_0$ is removed from $\mathbf{t}$ for optimization reasons. It is known that $\mathbf{t}_0$ can be recovered from multiple signatures [AOVCG25]. The number of signatures was quantified by Azevedo-Oliveira et al. [AOVCG25], who showed that 200 000 signatures are sufficient to recover $\mathbf{t}_0$ for ML-DSA-44 while 500 000 signatures are sufficient for ML-DSA-65 and ML-DSA-87.

Table 2: ML-DSA parameters relevant to key generation.

Parameter set	(k, ℓ)	η	q	d
ML-DSA-44	(4,4)	2	8380417	13
ML-DSA-65	(6,5)	4	8380417	13
ML-DSA-87	(8,7)	2	8380417	13

2.5.2 Secret Key Unpacking

The vectors $\mathbf{s}_1$ and $\mathbf{s}_2$ are stored in the secret key sk in an encoded form consisting of a sequence of packed bytes. For ML-DSA-65, each coefficient of $\mathbf{s}_1$ and $\mathbf{s}_2$ is encoded using 4 bits, allowing the nine possible coefficient values to be represented compactly. During signing, the implementation decodes the packed bytes and reconstructs the corresponding coefficients using a secret key unpacking procedure[1].

The secret key unpacking procedure creates two complementary representations of the coefficients of $\mathbf{s}_1$ and $\mathbf{s}_2$: packed and unsigned in the range $[0, 2\eta]$ and unpacked and signed in the range $[-\eta, \eta]$ (described in more detail in Section 6). The attack presented in this paper utilizes the complementary leakage from these two representations to improve classification accuracy and detect misclassified coefficients during secret key recovery.

2.6 Syndrome Decoding

Syndrome decoding is a classical problem in coding theory that finds, for a given linear code and a received word, a low-weight error pattern that corresponds to the deviation from a valid codeword. Let $\mathcal{C}$ be a linear code with parity-check matrix $\mathbf{H}$, such that every codeword $\mathbf{x} \in \mathcal{C}$ satisfies $\mathbf{Hx} = \mathbf{0}$. Given a received word $\mathbf{r} = \mathbf{x} + \mathbf{e}$ affected by an unknown error vector $\mathbf{e}$, the *syndrome* is

$$\mathbf{s} = \mathbf{Hr} = \mathbf{H}(\mathbf{x} + \mathbf{e}) = \mathbf{He}. \tag{5}$$

The syndrome therefore depends only on the error $\mathbf{e}$ and not on the underlying codeword $\mathbf{x}$. Syndrome decoding consists of finding an error vector $\hat{\mathbf{e}}$ satisfying

$$\mathbf{H}\hat{\mathbf{e}} = \mathbf{s}, \tag{6}$$

typically with minimum Hamming weight $\mathrm{wt}(\mathbf{e})$. The general syndrome decoding problem for linear codes is NP-complete [BMvT78].

Information-set decoding (ISD) is a family of randomized algorithms used to solve the syndrome decoding problem for linear codes [Pra62]. In this paper, Stern's ISD algorithm [Ste89] is used as a computational tool within the key recovery method described in Section 7.1. The positions of coefficients of $\mathbf{s}_1$ and $\mathbf{s}_2$ that might have been misclassified by the distinguisher are interpreted as an error vector. The known structural constraints on the secret key restrict the possible error values, letting us define the problem as a version of a restricted syndrome decoding problem [BPW+23].

3 Related Work

This paper describes prior work on GMM-based and horizontal attacks, profiled and non-profiled single-trace attacks against PQC implementations, and non-profiled attacks against ML-DSA.

[1] It is worth mentioning that a similar secret key unpacking procedure is used in other PQC algorithms, including ML-KEM [Nat24b].

3.1 GMM-based and Horizontal Attacks

GMMs have long been used to model side-channel leakage, both in profiled and non-profiled settings. Chari et al. [CRR03] introduced template attacks, in which multivariate GMMs are constructed during a profiling phase to characterize the leakage associated with target intermediate values. Lemke-Rust and Paar [LRP07] subsequently applied GMMs to higher-order side-channel analysis of masked implementations, demonstrating that mixture models can be fitted even when some information, such as random masks, is unavailable.

GMMs have also been combined with horizontal attacks, where multiple observations are extracted from a single execution. Perin and Chmielewski [PC16] used clustering techniques, including GMMs, to attack a blinded RSA implementation on an STM32F4 with a single trace. Their approach identifies repeated loop iterations within the trace and uses the noisy classifications as input to an iterative attack that refines them. More recently, Cler et al. [COM23] showed that fitting a two-class GMM to segmented traces can be used for identifying POIs in horizontal attacks against RSA and ECC implementations. These approaches primarily exploit leakage that distinguishes between two different execution paths.

GMMs have also been used directly as non-profiled distinguishers. Béguinot et al. [BCGR22] introduced a non-profiled attack that estimates leakage model parameters directly from attack traces. Their approach can be interpreted as an adaptation of a maximum-likelihood distinguisher. It requires a few hundred traces to recover the key from a first-order low-entropy masked AES implementation running on an 8-bit ATMega163 smart card. Qian et al. [QGQ+25] combined GMM-based clustering with neural-network refinement to perform non-profiled single-trace attacks against ECC, RSA, and SM2 implementations. Their initial GMM provides pseudo-labels that are subsequently refined by neural networks.

These works demonstrate that GMMs and horizontal analysis can substantially reduce the dependence on profiling. However, they address different aspects of the problem from the one considered in this paper. Existing GMM-based horizontal attacks primarily use clustering to identify two execution paths or to initialize a subsequent learning procedure, while existing non-profiled GMM attacks generally use multiple traces to distinguish among classes. In contrast, we use a non-profiled GMM to distinguish among multiple classes from a single trace.

3.2 Profiled Single-Trace Attacks against PQC Implementations

Single-trace profiled attacks have been demonstrated against a wide range of PQC algorithms, including ML-DSA [WNGD24, HLK+21], ML-KEM [PPM17, PP19, JNWD24], SABER [SKL+20], FALCON [QA26], HQC [DGN26], UOV [ACK+23], MAYO [JD26], and FrodoKEM [SKL+20]. These attacks employ profiled distinguishers, such as templates or neural networks, which are created using traces acquired during a profiling phase.

The work most closely related to ours is that of Wang et al. [WNGD24] where a single-trace deep learning-based side-channel attack against the secret-key unpacking (the first step of the signing procedure) of ML-DSA is presented. The attack exploits complementary leakage from the unsigned packed and the signed unpacked representations of the coefficients of the secret vectors $\mathbf{s}_1$ and $\mathbf{s}_2$. By classifying the HWs of both representations, the neural networks recover each coefficient by mapping the resulting HW pair to a unique value. The attack is demonstrated on an STM32F4 implementation of ML-DSA-44 running at 24 MHz. It can recover the full secret key with a 9 % success rate without enumeration, assuming that $\mathbf{t}_0$ is known [WNGD24].

We target the same secret key unpacking procedure and exploit the same complementary leakage from the unsigned and signed representations as in the attack by Wang et al. [WNGD24]. However, their attack relies on a profiled deep-learning classifier, whereas

our GMM distinguisher is entirely non-profiled. Moreover, their attack is demonstrated on ML-DSA-44, while ours targets ML-DSA-65 for which coefficients within a larger range of possible values must be recovered, making classification more challenging.

3.3 Non-profiled Single-Trace Attacks against PQC Implementations

We are aware of two non-profiled single-trace attacks against PQC implementations that do not depend on soft-analytical post-processing. In [SKL+20], a side-channel attack on ML-KEM using binary k-means as a distinguisher is presented. In [GR24], a side-channel attack on HAWK using a threshold to distinguish between two classes is described. The latter attack also classifies ≈1% of the coefficients using ternary k-means, although this additional classification is not essential for the attack's success.

Both attacks target implementations running on the STM32F3 processor, which is closely related to the STM32F4 processor used in our experimental setup. However, they rely on substantially stronger leakage. As shown in Table 1, the minimum ΔHW values for the attacks [SKL+20] and [GR24] are 16 and 17, respectively.

3.4 Non-profiled Multi-Trace Attacks against ML-DSA

Most non-profiled multi-trace side-channel attacks against Dilithium and ML-DSA target the pointwise multiplication $\hat{c} \circ \hat{\mathbf{s}}_1$ in the Number Theoretic Transform (NTT) domain. Chen et al. [CKA+21] presented a CPA attack against Dilithium-2, recovering its secret key from 157 traces. Liu et al. [LLZ+24] further improved the CPA efficiency using the same operation as an attack point. Qiao et al. [QLZ+23] combined CPA with a Short Integer Solution (SIS) solver and demonstrated key recovery for Dilithium-2, 3, and 5. Tosun and Savas [TS24] reduced the number of attack traces through zero-value filtering, reaching 220 traces for Dilithium-3, and demonstrated key recovery from 35 traces for an unoptimized (`-O0`) implementation on an STM32F303 [TOS25]. The latter work also considers higher-order non-profiled attacks against masked implementations.

In all these attacks, each secret coefficient of $\mathbf{s}_1$ is processed only once in the targeted pointwise multiplication within a given signature, limiting the amount of information that can be extracted about an individual coefficient from a single execution.

A recent work by Zhao et al. [ZHC+26] further reduces the number of attack traces by exploiting a different attack point. They target the matrix-vector product $\hat{\mathbf{w}} = \hat{\mathbf{A}} \circ \hat{\mathbf{y}}$ computed during signing. Because the ephemeral vector $\hat{\mathbf{y}}$ is reused across the rows of the public matrix $\hat{\mathbf{A}}$, a single signing execution contains multiple leakage observations associated with each ephemeral coefficient. Combined with the relation $\hat{\mathbf{y}} = \hat{\mathbf{z}} - \hat{c} \circ \hat{\mathbf{s}}_1$, these observations yield hypotheses on $\mathbf{s}_1$. A non-profiled CPA distinguisher is used and information from several leakage points is combined using a weighted-fusion technique, followed by an algebraic procedure that exploits the small-norm structure of $\mathbf{s}_1$. On an STM32F405, they report full recovery of $\mathbf{s}_1$ from 6, 4, and 4 traces for ML-DSA-44, ML-DSA-65, and ML-DSA-87, respectively, in the unprotected setting.

In our work, rather than exploiting repeated multiplication by public values, we target the secret-key unpacking procedure. Instead of accumulating correlation information over multiple executions, we exploit the processing of secret coefficients within a single execution and analyze the resulting observations jointly as a distribution using a GMM.

4 Threat Model

We assume that the attacker has temporary physical access to a device that performs signing using an unknown secret key. The attacker knows the algorithm specification, the public key, and other non-secret parameters (such as $\mathbf{t}_0$ for ML-DSA). The attacker

can observe valid signatures and measure the device's power consumption during signing operations.

The attacker's goal is to recover the secret key, or sufficient information required to fully reconstruct it, from a single power trace.

We consider a black-box scenario in which the attacker has no detailed knowledge of the target implementation beyond the algorithm specification. We assume that the implementation does not employ side-channel countermeasures to protect the secret key unpacking procedure.

Furthermore, we assume that the attacker has basic knowledge of signal processing and power-measurement techniques and can make minor modifications to the target device's hardware to facilitate power measurements. These modifications do not change the functional behavior of the algorithm running on the target device.

5 Noise Reduction

This section describes techniques which we apply to lower the noise floor and improve the distinguishability of the leakage.

5.1 Signal Preprocessing

We preprocess traces by applying a third-order Butterworth bandpass filter using both a forward and backward pass. The bandpass filter suppresses transient high-frequency measurement distortions and low-frequency drift causing vertical misalignment while preserving the frequency components containing the leakage concentrated around the system-clock frequency. The choice of the lower and upper cutoff frequencies is important for maintaining class separability.

The cutoff frequencies can be selected by exhaustively enumerating a small set of candidate values during the attack stage. For example, using eight candidates for the lower cutoff frequency and eight candidates for the upper cutoff frequency results in $8 \times 8 = 64$ candidate filters. If complementary leakage representations are available, a more efficient method for selecting cutoff frequencies, presented in Section 6.4, can be used.

Higher sampling rates increase the number of clean samples available between transient disturbances, improving the filter's ability to suppress them. In our experiments with the STM32F4 running at 6 MHz, 40-fold oversampling is sufficient for effective filtering.

5.2 Measurement Setup Modifications

In our experiments, we use a ChipWhisperer HuskyPlus with a CW308 UFO expansion board for power side-channel acquisition. The target device is a CW308T-STM32F4HWC board equipped with an ARM Cortex-M4-based STM32F415RGT6 microcontroller.

Below we describe minor modifications to the target hardware which we performed. Each modification addresses a distinct source of noise identified during setup characterization. While presented here in terms of our specific equipment, the underlying principles (targeting common-mode noise, power supply-induced interference, and sampling resolution) may apply to other power measurement setups facing similar noise sources.

Differential probing. The Husky can perform differential measurements, however the CW308 UFO expansion board does not have physical ports to support them. To make use of this Husky feature, we solder two short ($\approx$4 cm) coaxial cables directly to the target board across the shunt resistor, with the cable shields grounded as close to the shunt as possible. This improves common-mode noise suppression and reduces interference from the longer, unshielded signal path on the expansion board. Differential probing also increases

the effective dynamic range by directly measuring the small voltage variations associated with the leakage.

Low ripple power supply. We replace the Husky's internal switching supply with an external low-ripple supply. The Husky's internal 3.3 V switching supply introduces a sinusoidal interference component at ≈340 kHz which is not sufficiently suppressed by differential measurements. Although the Husky's switching regulator typically operates at ≈1 MHz, the observed frequency may result from step skipping at low load or from filtering of all but the second harmonic. Supplying the target through the expansion board's external DC input with a cleaner power source eliminates the observed interference.

Reduced shunt resistance. We reduce the shunt resistance from the original 10 Ω to 1 Ω. With the original resistance, sharp variations in the measured voltage produce oscillations that can persist over several clock cycles, obscuring the instantaneous leakage. We found that lowering the resistance to 1 Ω (half of the maximum equivalent series resistance (ESR) specified for the decoupling capacitor in the datasheet [STM26]), suppresses most of the oscillations. We hypothesize that the observed oscillations are caused by instability of the internal low-dropout regulator (LDO) supplying the 1.2 V core voltage on the core voltage decoupling pins, to which the shunt is connected. We observe that the oscillations are independent of the clock frequency and therefore affect fewer clock cycles at a lower frequency.

Core overvolting. We increase the externally injected core voltage (at the far end of the shunt) from 1.2 V to 1.3 V. The higher core voltage reduces the remaining oscillations and increases the dynamic power consumption, potentially improving leakage distinguishability. We tried even higher voltages, but observed no further improvement and therefore settled at 1.3 V. Higher voltage may damage the target board, whose nominal core voltage is 1.26 V and maximum rated voltage is 1.32 V [STM26]. Increasing the externally provided core voltage is recommended by the board's manufacturer [New25] as a means to overpower the internal voltage regulator.

Slowed target clock. We replace the STM32F4's external clock signal with a 6 MHz clock signal supplied by our measurement setup. The low clock frequency enables high oversampling and reduces the effects of low-pass filtering and temporal blurring in the measurements. At the Husky's maximum sampling rate of 250 MHz, we get 40 samples per clock cycle for our experiments.

5.3 Noise Quantification

Next, we quantify the effectiveness of the techniques described above in reducing noise. As a metric, we use the normalized standard deviation, σ_{norm}, introduced in [KPP20]. Under the HW leakage model in eq. (2), σ_{norm} corresponds to scaling the model so that $a = 1$, $b = 0$, and $\epsilon \sim N(0, \sigma^2_{norm})$, and estimating σ_{norm} from the observed samples in each HW class. The normalized metric provides a convenient way to compare leakage across devices and to parametrize simulated attacks. Unlike the signal-to-noise ratio (SNR), it does not depend on the distribution of the leakage variable, but characterizes the noise relative to a unit change in HW.

For the three main POIs in our attacks, we calculate σ_{norm} as the mean of the standard deviation of all classes after scaling. We compare the following three cases: the default setup, the default setup with filtering applied, and the modified setup with filtering applied. We run the default setup at the same 6 MHz as the modified setup to enable the same level of oversampling.

Figure 1 shows how σ_{norm} is reduced when multiple traces representing repeated executions are averaged. We can see that, in the middle and right plots, σ_{norm} of a single trace in the modified setup with filtering applied is equal to that after averaging 7 and 29 traces in the default setup, respectively. For the filtered-only setup, 2 traces have to be

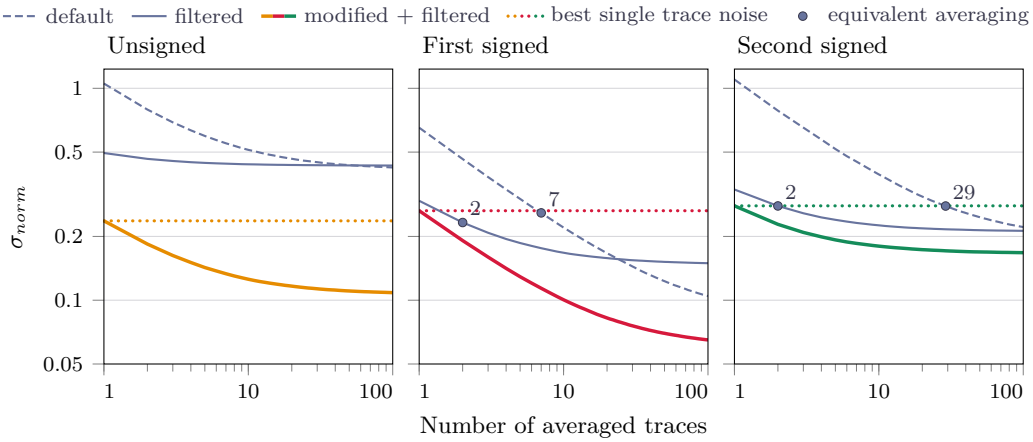


Figure 1: The effect of averaging on σ_{norm} for the default setup, the default setup with filtering applied, and the modified setup with filtering applied. The dots indicate the number of averaged traces required to reduce the noise to σ_{norm} of a single trace in the modified setup with filtering applied.

averaged to obtain the same σ_{norm}. In the left plot, however, σ_{norm} of a single trace in the modified setup with filtering applied is never reached, even when averaging 100 traces.

Achieving the low σ_{norm} in a single trace in the modified setup with filtering applied enables $\Delta\text{HW} = 1$ classification in the presented attack.

6 Key Recovery Attack on ML-DSA-65 Implementation

We demonstrate the new attack method on the latest ML-DSA-65 implementation from the *pqm4* PQC library [KPR+] optimized for Cortex-M4. The implementation is compiled at the highest optimization level, `-O3`. We verified that the resulting assembly is identical to that of the default *pqm4* firmware build, up to address offsets. As mentioned earlier, the attack does not require knowledge of the firmware: the information available from the algorithm specification [Nat24a], combined with minimal side-channel reverse engineering, is sufficient to identify the relevant operations.

6.1 Trace Segmentation

We first collect power traces containing the execution of the secret key unpacking procedure. This procedure is straightforward to locate because key unpacking is the first step of the signing algorithm specified in [Nat24a]. At the beginning of the traces, we identify a pattern that repeats $\ell + k$ times, once for each polynomial. Each occurrence in turn reveals a subpattern that repeats 128 times. This gives us the number of iterations in the unpacking loop. Since each polynomial in $\mathbf{s}_1$ and $\mathbf{s}_2$ has 256 coefficients, we conclude that each loop iteration processes two coefficients and segment the traces into iterations.

We then bandpass-filter the traces using $8 \times 8 = 64$ cutoff frequency combinations. Given the system clock frequency of 6 MHz, the highpass cutoff frequency, f_{hp}, ranges from 0.5 MHz to 4 MHz, while the lowpass cutoff frequency, f_{lp}, ranges from 5 MHz to 8.5 MHz, both with increments of 0.5 MHz. For each cutoff frequency pair, we perform the complete attack and select an optimal (f_{hp}, f_{lp}) pair using the metric from eq. (8) introduced in Section 6.4. Because the filtering is performed in software, the cutoff frequencies can

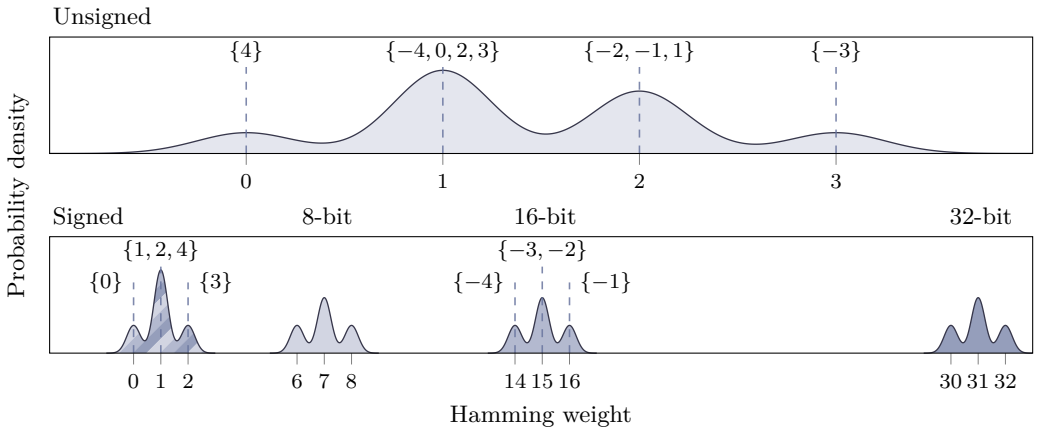


Figure 2: Expected HW leakage distributions for signed and unsigned representations at 8-, 16- and 32-bit widths.

be treated as postprocessing parameters and do not need to be selected before trace acquisition.

For each sample point within the segments, we subtract its mean over the 128 segments of the trace, so that the distribution is centered at zero.

6.2 POI Localization

In the *pqm4* ML-DSA implementation [KPR+], the secret key unpacking procedure is realized by the function `small_polyeta_unpack`. It corresponds to $BitUnpack(v, a, b)$ in the FIPS-204 specification [Nat24a], where v is a byte array and $a = b = \eta$. The function unpacks a secret key polynomial whose coefficients are stored in a packed byte array, with each coefficient represented by four bits. This agrees with our observation from power traces that two coefficients are processed per iteration. In the instruction set of Cortex-M4, the smallest number of packed coefficients that can be loaded at once is two, using a byte-load instruction. The packed unsigned coefficients c_i^u in the range $[0, 2\eta]$ are converted to their unpacked signed representation c_i^s in the range $[-\eta, \eta]$ according to

$$c_i^s = \eta - c_i^u.$$

An attacker may expect leakage from both the signed and unsigned representations.

We estimate the expected leakage distributions by constructing GMMs for both representations. Figure 2 shows the resulting PDFs, evaluated using eq. (3), for bitwidths of 8, 16 and 32 bits. For the signed representation, the bitwidth determines the separation between the positive and negative coefficient clusters.

For each sample point within an iteration, we fit separate GMMs to the signed and unsigned representations and use the resulting model fit scores to identify promising POIs. The GMMs use the prior probabilities of the coefficients' HWs as fixed weights w_k, variable equidistant means μ_k and a shared σ. For the subsequent classification step, we instead fit GMMs with free-moving means to maximize classification accuracy. Model fitting is done using expectation maximization with regards to the observed samples from a single trace.

At each sample point, we compute the log-likelihood ratio between the fitted model and a single-Gaussian null hypothesis evaluated on the observed data, as shown in Figure 3. We exclude sample points for which any predicted class occurs fewer than one third of its expected number of times. We perform this test on the unfiltered traces, as filtering reduces noise but also spreads the leakage over neighboring samples. Note that the clock cycles

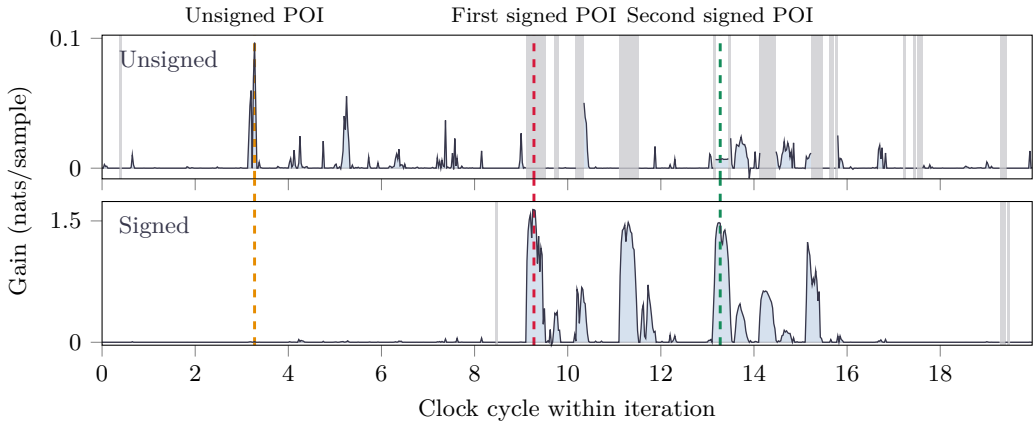


Figure 3: Locating POIs using log-likelihood scores of the fitted signed GMMs.

Table 3: Average log-likelihood scores of the fitted signed GMM.

POI	8-bit	16-bit	32-bit	Selected
First	2.95	**3.51**	3.36	16-bit (100%)
Second	3.01	**3.50**	3.39	16-bit (100%)

are not synchronized with the beginning of the iteration because the trace segmentation is based solely on repeating patterns in the trace.

In Figure 3, for the signed representation, there are four repeated patterns of peaks separated by two clock cycles. Because the ARMv7E-M store instruction takes two clock cycles, we hypothesize that pairs of these patterns correspond to the beginning and end of each memory transaction. We select the first peak of each pair (marked by a dashed line in the bottom plot of Figure 3) as a POI as it has the highest ratio of the two. In this way, we identify two POIs corresponding to the signed representations of the two coefficients processed in an iteration. However, for the unsigned representation, there is only one pair of peaks (marked by a dashed line in the top plot of Figure 3). At this stage, we do not know whether this unsigned representation corresponds to the first or to the second coefficient processed in an iteration. Its correspondence is resolved in Section 6.4.

As we mentioned earlier, the bitwidth affects the separation between the positive and negative coefficient clusters in the signed leakage. The bitwidth is initially unknown. We determine it by comparing the log-likelihood of the fitted model for each candidate bitwidth with the observed samples at the two signed POIs. Table 3 lists the average scores computed over non-averaged traces from 1000 key unpacking executions. We can see that the 16-bit model has the highest scores for both POIs in every trace.

Listing 1 shows the assembly of `small_polyeta_unpack` with the three identified POIs highlighted. Each of them corresponds to a 16-bit store instruction. Note that the presented POI identification method does not require the assembly. We present the assembly here purely to point out the leakage source in the ML-DSA implementation.

6.3 HW Classification and Coefficient Recovery

Figure 4 shows the observed sample distributions at the three identified POIs. We subsequently fit the GMM to the observations using approximated class means as initialization values. The fitted HW class means are marked in the figure.

```
1  // void small_polyeta_unpack(smallpoly *r, const uint8_t *a)
2      push        {lr}
3      add.w       ip, r1, #0xffffffff
4      movs        r2, #0
5      add.w       lr, r0, #2
6  .loop
7      ldrb.w      r3, [ip, #1]!
8      and.w       r3, r3, #0xf
9      strh.w      r3, [r0, r2, lsl #2]          ◀ Unsigned POI
10     ldrb.w      r1, [ip]
11     rsb         r3, r3, #4
12     strh.w      r3, [r0, r2, lsl #2]          ◀ First signed POI
13     lsrs        r3, r1, #4
14     rsb         r3, r3, #4
15     strh.w      r3, [lr, r2, lsl #2]          ◀ Second signed POI
16     adds        r2, #1
17     cmp         r2, #0x80
18     bne.n       .loop
19     ldr.w       pc, [sp], #4
```

Listing 1: Assembly of the `small_polyeta_unpack` procedure compiled at `-O3`.

For each coefficient, the sample is assigned to the GMM component with the highest posterior probability. The posterior probability is additionally used as a confidence measure.

Let s_i^u and s_i^s denote the samples at the unsigned and signed POI of an ith coefficient, respectively. We determined earlier that the unsigned POI is available for every other index only. Let $\widehat{hw}_i^{\,a}$ and γ_i^a denote the HW classification and posterior probability of s_i^a, respectively, and $P(j \mid s_i^a)$ denote the posterior probability of class j given sample s_i^a, where $a \in \{u, s\}$. We calculate $\widehat{hw}_i^{\,a}$ and γ_i^a as (the superscript a is omitted for brevity):

$$P(j \mid s_i) = \frac{w_j \mathcal{N}(s_i \mid \mu_j, \sigma^2)}{\sum_p w_p \mathcal{N}(s_p \mid \mu_p, \sigma^2)}, \qquad \widehat{hw}_i = \arg\max_j P(j \mid s_i), \qquad \gamma_i = \max_j P(j \mid s_i).$$

where $j, p \in \{1, 2, \ldots, K\}$ are the class indexes. The relative frequency of classified HWs is compared against the expected distribution of HWs for the coefficients. A close match confirms that the POIs leak information about the targeted coefficients.

If only the signed POI is used for coefficient classification, the average number of coefficients that can be uniquely determined from $\widehat{hw}_i^{\,s}$ is $4/9 \approx 44.4\,\%$. Additionally considering $\widehat{hw}_i^{\,u}$ for every other coefficient (for which leakage from the unsigned representation is available) increases the fraction to $72.2\,\%$. This improvement is due to the fact that the HW pairs composed of signed and unsigned representations of coefficient values $c \in \{-\eta, \ldots, \eta\}$ induce a unique mapping $\varphi(c)$ given by:

$$c \mapsto (HW(\eta - c), HW(c)), \tag{7}$$

where $HW(x)$ is the HW of the 16-bit two's-complement representation of x. Since φ is injective over $\{-\eta, \ldots, \eta\}$, each valid coefficient value c yields a distinct pair of HW values. The inverse φ^{-1} is defined only partially, since not every possible HW classification pair $(\widehat{hw}_i^{\,u}, \widehat{hw}_i^{\,s})$ corresponds to a valid coefficient value.

We show in Section 7 that $72.2\,\%$ of all coefficients are more than sufficient to reconstruct the full secret vectors $\mathbf{s}_1$ and $\mathbf{s}_2$. For this reason, we commit only the most confident classifications and disregard the rest. Throughout the rest of the paper, we refer to the

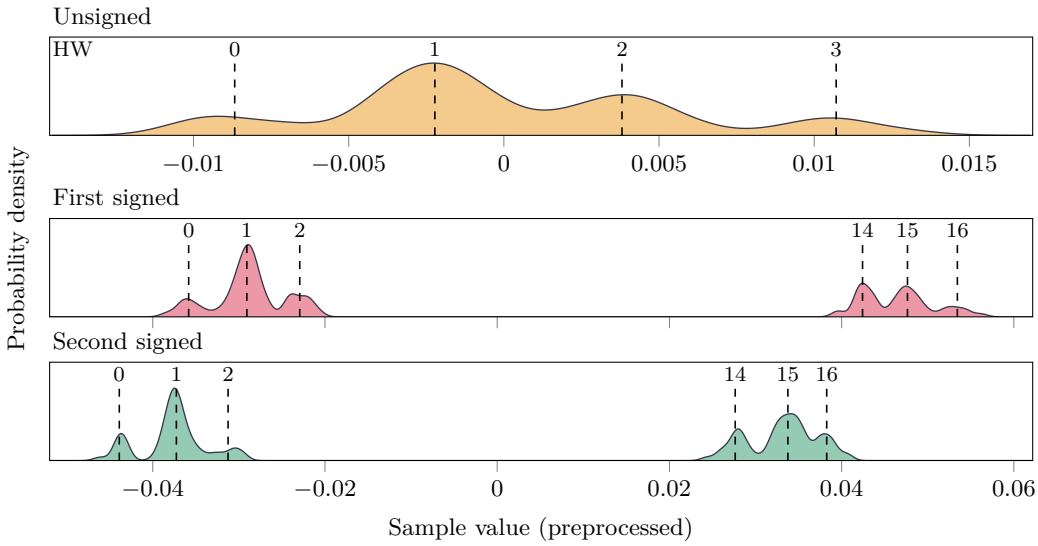


Figure 4: Sample distributions at the three POIs over 128 iterations representing the unpacking of one polynomial of a secret vector by the secret key unpacking procedure. The class means identified by the distinguisher are highlighted.

coefficients whose classifications are accepted as *committed coefficients*. Clearly, some committed coefficients may be erroneous.

For indices i for which both signed and unsigned representations are available, we use the lower of the two confidence values. Otherwise, we use the confidence value of the signed representation. In other words, the confidence score is defined as

$$\gamma_i = \begin{cases} \gamma_i^s, & \text{if only the signed representation is available,} \\ \min\left(\gamma_i^u, \gamma_i^s\right), & \text{if both representations are available.} \end{cases}$$

The coefficients are ranked according to this score.

6.4 Exploiting Complementary Leakage

Complementary leakages of the unsigned and signed representations allow us to associate the unsigned POI with one of the two signed POIs and select the optimal filtering frequencies.

In a non-profiled setting, in which the true coefficient values are unknown, this can be done by counting the number of *detectable misclassifications*. Let $\mathcal{I}_{\text{dual}} \subseteq \{1, \ldots, N_{\text{coeffs}}\}$ denote the set of coefficient indices for which both unsigned and signed representations are available. We derive a lower bound on the total number of misclassifications as the sum of *inadmissible* HW pairs which are mapped to no valid coefficient by the decoding (7):

$$N_{\text{mis}} = \left| \left\{ i \in \mathcal{I}_{\text{dual}} : \left(\widehat{hw}_i^{\,\text{u}}, \widehat{hw}_i^{\,\text{s}}\right) \notin \operatorname{im} \varphi \right\} \right|. \tag{8}$$

N_{mis} is a lower bound because misclassifications that produce another valid HW pair and misclassifications where $i \notin \mathcal{I}_{\text{dual}}$ are not included in the count.

By computing N_{mis} for each of the two signed POIs, we determine that the unsigned POI corresponds to the first of the two signed POIs, as the number of misclassifications for this case is significantly lower. Consequently, $\mathcal{I}_{\text{dual}} = \{ i \in \{1, \ldots, N_{\text{coeffs}}\} : i \text{ odd} \}$. In our experiments, the first POI is consistently chosen with a significant margin.

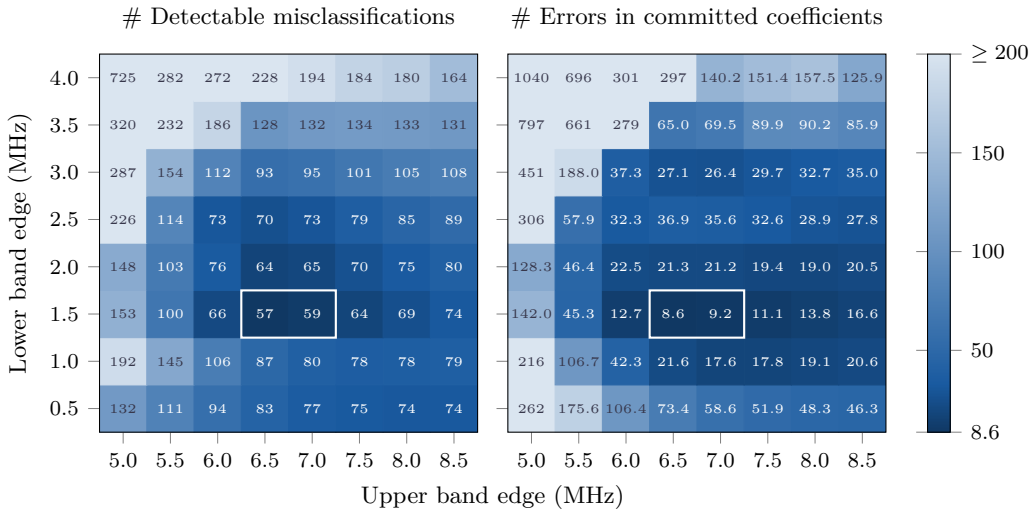


Figure 5: Selecting a bandpass cutoff frequency pair yielding the minimal number of detectable misclassifications. The right-hand side shows the number of errors in the committed coefficients computed empirically.

We select the optimal bandpass cutoff frequencies as the pair (f_{hp}, f_{lp}) that yields the smallest number of detectable misclassifications. As illustrated in Figure 5 (left), two pairs: $(f_{hp}, f_{lp}) = (1.5\,\text{MHz}, 6.5\,\text{MHz})$, and $(f_{hp}, f_{lp}) = (1.5\,\text{MHz}, 7.0\,\text{MHz})$ show the lowest N_{mis}. Figure 5 (right) shows that the actual number of errors in the committed coefficients correlates with N_{mis}.

7 From Partial to Full Key Recovery

If a sufficiently large part of the coefficients of secret vectors $\mathbf{s}_1$ and $\mathbf{s}_2$ are known, it is possible to recover the remaining coefficients by solving the system of linear equations given by eq. (4). This method, first presented in [WNGD24], assumes knowledge of the least significant bits of $\mathbf{t}$.

Since for ML-DSA-65 $(k, \ell) = (6, 5)$ and $n = 256$, eq. (4) induces a system of $n \times k = 1536$ linear equations with $n \times (\ell + k) = 2816$ unknown variables. In order to solve this system, we have to know at least $n \times \ell = 1280$ coefficients of $\mathbf{s}_1$ and $\mathbf{s}_2$, i.e. $\ell/(\ell+k) = 5/11 \approx 0.455$ of all. For signature forgery, it is sufficient to know $\mathbf{s}_1$, though to recover $\mathbf{s}_1$ using the linear equations-based method, we must solve for both secret vectors.

Consequently, as long as we can correctly determine $1280 - E$ coefficients of $\mathbf{s}_1$ and $\mathbf{s}_2$ by side-channel analysis, where E is the number of errors in the committed coefficients, and enumerate the E misclassified coefficients, we can fully recover $\mathbf{s}_1$ and $\mathbf{s}_2$. Recall from Section 6.3 that, on average, we can recover 72.2 % of all coefficients using both the signed and unsigned representations, corresponding to 2034 coefficients. This implies that $2034 - 1280 = 754$ least confident classifications can be discarded.

It is worth mentioning that using lattice reduction for full key recovery is not a good option in our setting. Wang et al. [WNGD24] have shown that for ML-DSA-44, given 75 % known coefficients of $\mathbf{s}_1$, the estimated Core Shortest Vector Problem (Core-SVP) bit hardness required to recover the remaining coefficients of $\mathbf{s}_1$ by Block Korkin-Zolotarev (BKZ) lattice reduction algorithm [SE94] with blocksize $\beta = 160$ is $\alpha = 46.7$. The ML-DSA-65 case is likely to be harder due to its larger parameters (see Table 2).

7.1 Correcting Errors in Committed Coefficients with ISD

Errors in the committed coefficients can be corrected either by a brute-force enumeration of possible coefficient values, or by committing to more coefficients and then identifying and rejecting the erroneous ones. The latter can be faster for small E as it requires enumerating error positions only, rather than positions and substituted values simultaneously, at the cost of committing to more coefficients.

In this section, we introduce a new, more efficient method for correcting errors in imperfect classifications. We prove that a system of equations induced by eq. (4) can be reduced to a syndrome decoding problem. Therefore, errors in the committed coefficients can be corrected using efficient ISD algorithms.

Any polynomial $b = \sum_{i=0}^{n-1} b_i X^i \in R_q$ is uniquely defined by its coefficient vector $\text{vec}(b) = (b_0, \ldots, b_{n-1})^\top \in \mathbb{Z}_q^n$. A polynomial product $a \cdot b \in R_q$ is equivalent to $\text{vec}(a \cdot b) = \text{Rot}(a)\,\text{vec}(b) \in \mathbb{Z}_q^n$ where $\text{Rot}(a) \in \mathbb{Z}_q^{n \times n}$ is the negacyclic matrix

$$\text{Rot}(a) = \begin{pmatrix} a_0 & -a_{n-1} & \cdots & -a_1 \\ a_1 & a_0 & \cdots & -a_2 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1} & a_{n-2} & \cdots & a_0 \end{pmatrix}.$$

We can therefore define the block matrix $\widetilde{\mathbf{A}} = (\text{Rot}(A_{i,j}))_{i,j} \in \mathbb{Z}_q^{kn \times \ell n}$, where $A_{i,j} \in R_q$ is the polynomial indexed by row i and column j of the matrix $\mathbf{A} \in R_q^{k \times \ell}$ in eq. (4).

We further define the coefficient vectors $\tilde{\mathbf{s}}_1 = \left(\text{vec}(s_{1,1})^\top \ldots \text{vec}(s_{1,\ell})^\top\right)^\top \in \mathbb{Z}_q^{\ell n}$, $\tilde{\mathbf{s}}_2 = \left(\text{vec}(s_{2,1})^\top \ldots \text{vec}(s_{2,k})^\top\right)^\top \in \mathbb{Z}_q^{kn}$, and $\tilde{\mathbf{t}} = \left(\text{vec}(t_1)^\top \ldots \text{vec}(t_k)^\top\right)^\top \in \mathbb{Z}_q^{kn}$ where $s_{1,j}, s_{2,i}, t_i \in R_q$ denote the components of $\mathbf{s}_1, \mathbf{s}_2$, and $\mathbf{t}$ in eq. (4), respectively, for $i \in \{1, \ldots, k\}$ and $j \in \{1, \ldots, \ell\}$. Now we can rewrite (4) as

$$\begin{bmatrix} \widetilde{\mathbf{A}} & \mathbf{I} \end{bmatrix} \begin{pmatrix} \tilde{\mathbf{s}}_1 \\ \tilde{\mathbf{s}}_2 \end{pmatrix} = \tilde{\mathbf{t}}$$

where $\mathbf{I}$ is the $kn \times kn$ identity matrix. The result is a linear system $\mathbf{Mx} = \mathbf{y}$ of dimension $kn \times (k + \ell)n$ over $\mathbb{Z}_q$, with $\mathbf{M} = [\widetilde{\mathbf{A}}\ \mathbf{I}]$, $\mathbf{y} = \tilde{\mathbf{t}}$, and $\mathbf{x} = (\tilde{\mathbf{s}}_1^\top\ \tilde{\mathbf{s}}_2^\top)^\top$, where each entry of $\mathbf{x}$ takes values in $\{-\eta, \ldots, \eta\}$.

Next, we partition the set defined as the union of coefficients of $\tilde{\mathbf{s}}_1$ and $\tilde{\mathbf{s}}_2$ into two subsets: the committed set C of size $|C| = \ell n + m$ and the uncommitted set U of size $|U| = kn - m$, where the parameter m, represents the number of *extra* committed coefficients that overdetermine the system $\mathbf{Mx} = \mathbf{y}$. Let $\mathbf{x}_C, \hat{\mathbf{x}}_C \in \{-\eta, \ldots, \eta\}^{|C|}$ denote vectors of real and classified values of elements of C, respectively, and $\mathbf{e} = \mathbf{x}_C - \hat{\mathbf{x}}_C$ denote an unknown error vector with $\text{wt}(\mathbf{e}) \leq E$, where E is the number of errors in the committed coefficients. Finally, let $\mathbf{M}_C$ and $\mathbf{M}_U$ be the submatrices of $\mathbf{M}$ indexed by C and U respectively, and $\mathbf{x}_U$ be a vector of real values of elements of U.

Theorem 1 (Reduction to syndrome decoding)**.** *Assume* $\mathbf{M}$ *has full row rank and* $\mathbf{M}_U$ *has full column rank. Then, the problem of correcting errors in* $\hat{\mathbf{x}}_C$ *can be reduced to the q-ary syndrome decoding problem* $\mathbf{He} = \mathbf{s}$*, where the parity-check matrix* $\mathbf{H} \in \mathbb{Z}_q^{m \times (\ell n + m)}$ *with rank* m *and the syndrome* $\mathbf{s} \in \mathbb{Z}_q^m$ *are derived from* $\mathbf{A}$*,* $\mathbf{t}$ *and* $\hat{\mathbf{x}}_C$*. Every solution* $\mathbf{e}$ *uniquely determines* $\mathbf{x}_U$ *such that* $(\mathbf{x}_C, \mathbf{x}_U)$ *satisfies* $\mathbf{Mx} = \mathbf{y}$*.*

Proof. By partitioning $\mathbf{M}$ into $\mathbf{M}_C$ and $\mathbf{M}_U$ and substituting $\mathbf{x}_C = \hat{\mathbf{x}}_C + \mathbf{e}$ into $\mathbf{Mx} = \mathbf{y}$, we get

$$\mathbf{M}_C(\hat{\mathbf{x}}_C + \mathbf{e}) + \mathbf{M}_U\mathbf{x}_U = \mathbf{y}. \tag{9}$$

Let $\mathbf{P} \in \mathbb{Z}_q^{m \times kn}$ be a basis matrix of the left kernel of $\mathbf{M}_U$, i.e. $\mathbf{PM}_U = \mathbf{0}$. Since $\mathbf{M}_U$ has full column rank, $\mathbf{P}$ has m rows. Left-multiplying both sides of eq. (9) by $\mathbf{P}$ eliminates

the term $\mathbf{M}_U\mathbf{x}_U$ with unknown $\mathbf{x}_U$ and yields the syndrome decoding problem $\mathbf{He} = \mathbf{s}$ where $\mathbf{H}$ and $\mathbf{s}$ are defined by

$$\mathbf{H} := \mathbf{PM}_C \in \mathbb{Z}_q^{m\times(\ell n+m)}, \quad \mathbf{s} := \mathbf{P}(\mathbf{y} - \mathbf{M}_C\hat{\mathbf{x}}_C) \in \mathbb{Z}_q^m. \tag{10}$$

Clearly, $\mathbf{PM} = [\mathbf{H}\ \mathbf{0}]$, up to column order. Since $\mathbf{M}$ has full row rank, right-multiplication by $\mathbf{M}$ preserves rank. Therefore, $\operatorname{rank}\mathbf{H} = \operatorname{rank}\mathbf{P} = m$.

Since $\mathbf{PM}_U = \mathbf{0}$ and $\mathbf{M}_U$ has full column rank, $\operatorname{col}(\mathbf{M}_U) \subseteq \ker\mathbf{P}$, where $\operatorname{col}(\mathbf{M}_U)$ stands for the column space of $\mathbf{M}_U$ and $\ker\mathbf{P}$ stands for the kernel of $\mathbf{P}$. Furthermore, by the rank–nullity theorem and the fact that $\operatorname{rank}\mathbf{P} = m$, the dimension of $\ker\mathbf{P}$ is $kn - m$. Since $\mathbf{M}_U$ has full column rank and $kn - m$ columns, its dimension is also $kn - m$. Hence, by the equal-dimension theorem, $\ker\mathbf{P} = \operatorname{col}(\mathbf{M}_U)$.

From the above, we can conclude that $\mathbf{He} = \mathbf{s}$ implies $\mathbf{y} - \mathbf{M}_C(\hat{\mathbf{x}}_C + \mathbf{e}) \in \operatorname{col}(\mathbf{M}_U)$, which, by the injectivity of $\mathbf{M}_U$, in turn implies that, for each $\mathbf{e}$, there exists a unique $\mathbf{x}_U$ such that $(\mathbf{x}_C, \mathbf{x}_U)$ satisfies eq. (9).

□

The expected number of error vectors $\mathbf{e}$ with $\operatorname{wt}(\mathbf{e}) \le E$ which resolve the syndrome decoding problem defined by eq. (10) is $\sum_{w\le E}\binom{\ell n+m}{w}(2\eta)^w q^{-m}$, i.e. it decreases rapidly with the growth of m. The probability of an incorrect solution $\mathbf{e}$ resulting in $\mathbf{x}_U$ with entries not in the range of $\{-\eta,\dots,\eta\}$ is $((2\eta+1)/q)^{kn-m}$, which is negligible in our case.

For $\mathbf{M} = [\widetilde{\mathbf{A}}\ \mathbf{I}]$, the assumption on full row rank always holds. For $\mathbf{M}_U$, the full column rank is not guaranteed in general, but, for a large q, the probability of a deficient rank is negligible and decreases with greater m. Additionally, sets C and U can be intentionally chosen so that $\mathbf{M}_U$ has full column rank, which is verified when computing $\mathbf{P}$.

The syndrome decoding instance defined by eq. (10) is a variant of a restricted syndrome decoding problem [BPW+23, Problem 1] in which the error vector $\mathbf{e}$ (for our high classification accuracy) has low weight, $\operatorname{wt}(\mathbf{e}) \le E$, and each entry of $\mathbf{e}$ takes at most 2η different values. This instance corresponds to a code of length $\ell n + m$ and dimension ℓn. Each of the m extra committed coefficients contributes exactly one parity check. On one hand, the number of parity checks increases with m, lowering the decoding cost. On the other hand, larger m increases the number of potential errors in $\hat{\mathbf{x}}_C$.

To select an optimal m, we estimate the complexity of solving the instance defined by eq. (10) using Stern's ISD algorithm [Ste89] over $\mathbb{Z}_q$ [Pet10] with adaptions to the restricted error alphabet from [BPW+23], for which $E + \theta - 2p \le m \le kn$, where θ and p are algorithm parameters[2]. The only modification to the formulas in [Pet10, Section 4], following by [BPW+23, Lemma 3 and the subsequent remark], is that the two lists of partial error candidates are built using only admissible error values. So the factor $(q-1)^p$ in the per-iteration complexity is replaced by $(2\eta)^p$. The expected number of iterations is the inverse of the iteration success probability and it remains unchanged since it depends only on the error positions. Following [Pet10], we treat addition and multiplication alike, quantifying the complexity as operations in $\mathbb{Z}_q$. While the values of entries of $\mathbf{e}$ depend on $\mathbf{x}_C$ and $\hat{\mathbf{x}}_C$, the *number* of possible values *per position*, and consequently the complexity, does not. Restricting the error alphabet reduces the size of the lists from $\binom{\ell n/2}{p}(q-1)^p$ to $\binom{\ell n/2}{p}(2\eta)^p$. The reduction makes the case of $p = 2$ computationally feasible, requiring $\approx 200\,\mathrm{MB}$ of memory. Increasing p to 3, however, would require $\approx 340\,\mathrm{GB}$. We therefore focus our analysis on $p \le 2$, for which we find $p = 2$ to be optimal.

The advantage of restricted Stern's algorithm compared to straightforward algorithms based on substituting possible coefficient values, or on committing to more coefficients and then rejecting the erroneous ones, is illustrated in Figure 6. Note that the attacker does not need to know the best m, since the complexity curves for the restricted Stern's algorithm in Figure 6 are relatively flat.

[2]We renamed the original parameter ℓ of Stern's algorithm to θ to avoid the confusion with ML-DSA's parameter ℓ.

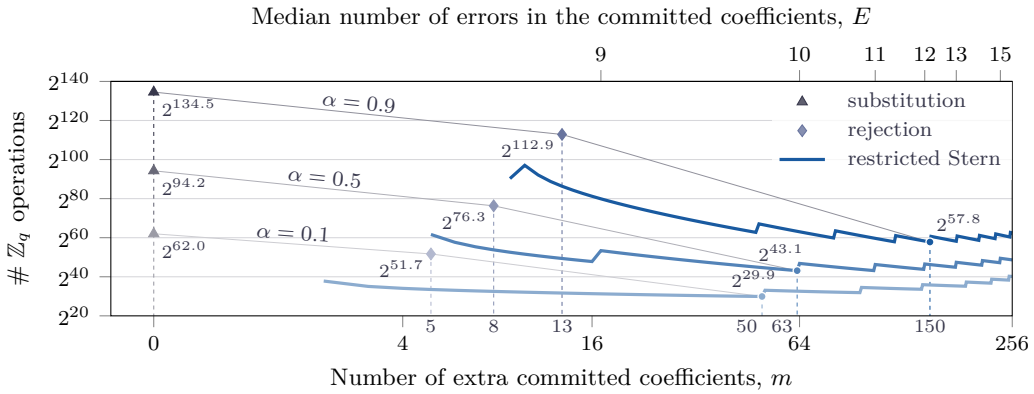


Figure 6: Computational complexities of restricted Stern's algorithm and substitution and rejection enumeration as functions of the number of extra committed coefficients m; the parameter α is the single-trace attack success probability.

8 Experimental Results

To quantify the success rate of the presented key recovery attack, we evaluate it on traces collected for 1,000 executions of the secret key unpacking procedure of ML-DSA-65, each performed with a different (sk, pk) pair selected at random.

First, all 256 coefficients of each of the $\ell = 5$ polynomials of $\mathbf{s}_1$ and $k = 6$ polynomials of $\mathbf{s}_2$ are recovered by power analysis. Then, independently for each of the 11 polynomials[3], the 256 coefficients are ranked by their confidence score γ, and at least the top-ranked $\lceil \frac{5}{11} \cdot 256 \rceil = 117$ coefficients are kept. This gives us sufficiently many coefficients to solve the system of linear equations induced by (4). As explained in the previous section, we select more than the minimum required number of top-ranked coefficients per polynomial to enable the syndrome decoding-based error correction of the committed coefficients.

Figure 7 shows the empirical probability of recovering the full secret key from a single trace (i.e., the fraction of 1,000 key recovery attempts that succeeded) using the method utilizing the restricted Stern's algorithm to correct errors in the committed coefficients as a function of the number of $\mathbb{Z}_q$ operations in the enumeration budget, for both the best m under an enumeration budget of 2^{70} and the frontier line representing the highest probability for all values of m.

We consider 2^{70} $\mathbb{Z}_q$ operations to be a conservative upper bound on the enumeration budget, following [ANS19] where the same number of algorithm executions is regarded enumerable. Within a budget of 2^{70} $\mathbb{Z}_q$ operations, the probability of a single-trace key recovery is 99 %. That drops to 93 % at a lower budget of 2^{60} and 71 % at a budget of 2^{50}, which is within the capability of a desktop computer. For a success probability of 50 %, the minimum required budget is $2^{43.1}$ $\mathbb{Z}_q$ operations.

9 Countermeasures

It has already been pointed out in [WNGD24] that the secret key unpacking in ML-DSA requires protection, and such advice should be considered for any secret handling operations. Masking [GP99, CJRR99] or shuffling [VCMKS12] are likely to be effective against the presented attack if implemented correctly.

[3]Sorting by confidence over all $256 \times 11 = 2816$ coefficients gives worse results in our experiments, as the GMMs are fitted individually per polynomial unpacking, resulting in incomparable confidence scores.

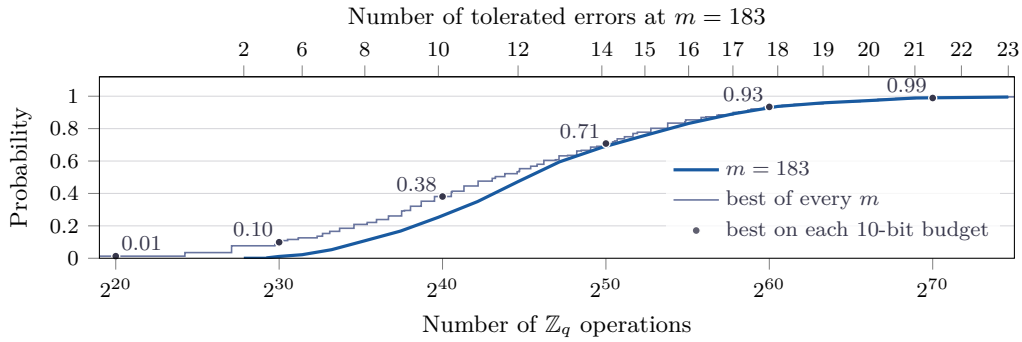


Figure 7: The empirical success probability of the single-trace full key recovery as a function of the number of $\mathbb{Z}_q$ operations in the enumeration budget. The case of $m = 183$ (the best m for 2^{70} operations) is plotted as a representative.

A simple ML-DSA implementation-specific adjustment that would increase the difficulty of the presented attack is flipping the sign convention for unpacking coefficients. In the specification [Nat24a], the unpacked coefficient is derived as $c^s = \eta - c^u$. If instead derived as $c^s = c^u - \eta$, the unique decoding mapping (7) would not exist. This change affects only the bit representation of the packed coefficients, not the coefficient values themselves.

The presented GMM-based HW classification method is tied to the physical design of the STM32F4 microcontroller. For this target, we show that, in some cases, the noise floor of store-instruction leakage can be reduced below the level at which averaging over repeated measurements saturates. When designing new processors, it could be of interest to intentionally scramble the power consumption associated with memory operations so that it does not linearly follow the HW model. While such an approach may not prevent the more advanced profiled attacks, it would make non-profiled, distribution-based classification significantly more difficult. It would also make locating leakage points in a black-box setting quite challenging.

10 Conclusion

We demonstrated that a non-profiled adversary without a clone device or implementation-specific knowledge can mount powerful single-trace attacks previously believed to require profiling. Our results show that the threshold for single-trace non-profiled attacks is considerably lower than previously assumed.

Future work includes automating POI identification and evaluating the presented horizontal distribution analysis method on other hardware targets and cryptographic implementations.

Acknowledgments

We would like to thank Mats Näslund for support and guidance that enabled this work. This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation and the Swedish Research Council (Grant No. 2025-05080).

References

[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 Transactions on Cryptographic Hardware and Embedded Systems*, 2023(4):58–79, Aug. 2023.

[ACK+23] Thomas Aulbach, Fabio Campos, Juliane Krämer, Simona Samardjiska, and Marc Stöttinger. Separating oil and vinegar with a single trace: Side-channel assisted Kipnis-Shamir attack on UOV. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2023(3):221–245, Jun. 2023.

[AEVR23] Guilhèm Assael, Philippe Elbaz-Vincent, and Guillaume Reymond. Improving single-trace attacks on the number-theoretic transform for Cortex-M4. In *2023 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pages 111–121, 2023.

[ANS19] ANSSI. Décision sur la sécurité algorithmique résiduelle ("Decision on remaining algorithmic strength"), 2019. Available: https://cyber.gouv.fr/documents/97/anssi-cc-note-23-remaining-strength_v1.01.pdf.

[AOVCG25] Paco Azevedo-Oliveira, Andersson Calle Viera, Benoît Cogliati, and Louis Goubin. Uncompressing Dilithium's public key. In *Advances in cryptology – CRYPTO 2025*, page 417–443, Berlin, Heidelberg, 2025. Springer-Verlag.

[BCGR22] Julien Béguinot, Wei Cheng, Sylvain Guilley, and Olivier Rioul. Side-channel expectation-maximization attacks. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2022(4):774–799, Aug. 2022.

[BCO04] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In Marc Joye and Jean-Jacques Quisquater, editors, *Cryptographic hardware and embedded systems - CHES 2004*, page 16–29, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[BMvT78] E. Berlekamp, R. McEliece, and H. van Tilborg. On the inherent intractability of certain coding problems (corresp.). *IEEE Transactions on Information Theory*, 24(3):384–386, 1978.

[BPW+23] Sebastian Bitzer, Alessio Pavoni, Violetta Weger, Paolo Santini, Marco Baldi, and Antonia Wachter-Zeh. Generic decoding of restricted errors. In *2023 IEEE International Symposium on Information Theory (ISIT)*, pages 246–251, 2023.

[CGL+24] Jean-Sébastien Coron, François Gérard, Tancrède Lepoint, Matthias Trannoy, and Rina Zeitoun. Improved high-order masked generation of masking vector and rejection sampling in Dilithium. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2024(4):335–354, Sep. 2024.

[CJRR99] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In Michael Wiener, editor, *Advances in cryptology — CRYPTO' 99*, page 398–412, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

[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, 2021.

[COM23] Gauthier Cler, Sebastien Ordas, and Philippe Maurine. Bernoulli at the root of horizontal side channel attacks. In *Smart Card Research and Advanced Applications: 22nd International Conference, CARDIS*, page 107–126, Berlin, Heidelberg, Nov. 2023. Springer-Verlag.

[CRR03] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In Burton S. Kaliski, Çetin Koç K., and Christof Paar, editors, *Cryptographic hardware and embedded systems - CHES 2002*, page 13–28, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[DGN26] Haiyue Dong, Qian Guo, and Denis Nabokov. Single-trace key recovery attacks on HQC using valid and invalid ciphertexts. In Joan Daemen and Emmanuel Thomé, editors, *Advances in Cryptology – EUROCRYPT 2026*, pages 568–597, Cham, 2026. Springer Nature Switzerland.

[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 Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):238–268, Feb. 2018.

[GP99] Louis Goubin and Jacques Patarin. DES and differential power analysis (the "duplication" method). In *Proceedings of the first international workshop on cryptographic hardware and embedded systems*, CHES '99, page 158–172, Berlin, Heidelberg, 1999. Springer-Verlag.

[GR24] Morgane Guerreau and Mélissa Rossi. A not so discrete sampler: Power analysis attacks on HAWK signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2024(4):156–178, Sep. 2024.

[HLK+21] Jaeseung Han, Taeho Lee, Jihoon Kwon, Joohee Lee, Il-Ju Kim, Jihoon Cho, Dong-Guk Han, and Bo-Yeon Sim. Single-trace attack on NIST round 3 candidate Dilithium using machine learning-based profiling. *IEEE Access*, 9:166283–166292, 2021.

[JD25] Yanning Ji and Elena Dubrova. A side-channel attack on a masked hardware implementation of CRYSTALS-Kyber. *Journal of Cryptographic Engineering*, 15(1):7, Apr 2025.

[JD26] Sönke Jendral and Elena Dubrova. Single-trace side-channel attacks on MAYO exploiting leaky modular multiplication. In *Proceedings of the 2025 1st Workshop on Quantum-Resistant Cryptography and Security*, QRSEC '25, page 21–30, New York, NY, USA, 2026. Association for Computing Machinery.

[JNWD24] Sönke Jendral, Kalle Ngo, Ruize Wang, and Elena Dubrova. Breaking SCA-protected CRYSTALS-Kyber with a single trace. In *2024 IEEE international symposium on hardware oriented security and trust (HOST)*, pages 70–73, 2024.

[KJJ99] Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael Wiener, editor, *Advances in cryptology — CRYPTO' 99*, page 388–397, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

[KPP20] Matthias J. Kannwischer, Peter Pessl, and Robert Primas. Single-trace attacks on Keccak. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020(3):243–268, Jun. 2020.

[KPR+] Matthias J. Kannwischer, Richard Petri, Joost Rijneveld, Peter Schwabe, and Ko Stoffelen. PQM4: Post-quantum crypto library for the ARM Cortex-M4. https://github.com/mupq/pqm4.

[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 *IEEE European Test Symposium, ETS 2024*, pages 1–6. IEEE, 2024.

[LRP07] Kerstin Lemke-Rust and Christof Paar. Gaussian mixture models for higher-order side channel analysis. In Pascal Paillier and Ingrid Verbauwhede, editors, *Cryptographic hardware and embedded systems - CHES 2007*, page 14–27, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[MOP07] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer, New York, NY, USA, 1 edition, 2007.

[MPP16] Houssem Maghrebi, Thibault Portigliatti, and Emmanuel Prouff. Breaking cryptographic implementations using deep learning techniques. In *Security, Privacy, and Applied Cryptography Engineering (SPACE 2016)*, pages 3–26. Springer International Publishing, 2016.

[Nat24a] National Institute of Standards and Technology. Module-lattice-based digital signature standard. Federal Information Processing Standards Publication FIPS 204, U.S. Department of Commerce, 2024.

[Nat24b] National Institute of Standards and Technology. Module-lattice-based key-encapsulation standard. Federal Information Processing Standards Publication FIPS 203, U.S. Department of Commerce, 2024.

[New25] NewAE Technology Inc. CW308T-STM32F. ChipWhisperer Documentation, 2025. Section “VCC-Int Supply”. Accessed: 2026-10-08.

[PC16] Guilherme Perin and Łukasz Chmielewski. A semi-parametric approach for side-channel attacks on protected RSA implementations. In Naofumi Homma and Marcel Medwed, editors, *Smart card research and advanced applications*, page 34–53, Cham, 2016. Springer International Publishing.

[Pet10] Christiane Peters. Information-set decoding for linear codes over $\mathbb{F}_q$. In *Proceedings of the Third International Conference on Post-Quantum Cryptography*, PQCrypto’10, page 81–94, Berlin, Heidelberg, 2010. Springer-Verlag.

[PK25] Eric Chun-Yu Peng and Markus G. Kuhn. Adaptive template attacks on the Kyber binomial sampler. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2025(3):470–492, Jun. 2025.

[PP19] Peter Pessl and Robert Primas. More practical single-trace attacks on the number theoretic transform. In Peter Schwabe and Nicolas Thériault, editors, *Progress in Cryptology – LATINCRYPT 2019*, pages 130–149, Cham, 2019. Springer International Publishing.

[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, *Cryptographic Hardware and Embedded Systems – CHES 2017*, pages 513–533, Cham, 2017. Springer International Publishing.

[Pra62] E. Prange. The use of information sets in decoding cyclic codes. *IRE Transactions on Information Theory*, 8(5):5–9, 1962.

[PZS17] Romain Poussier, Yuanyuan Zhou, and François-Xavier Standaert. A systematic approach to the side-channel analysis of ECC implementations with worst-case horizontal attacks. In Wieland Fischer and Naofumi Homma, editors, *Cryptographic Hardware and Embedded Systems - CHES 2017*, pages 534–554. Springer, 2017.

[QA26] Jinyi Qiu and Aydin Aysu. SHIFT SNARE: uncovering secret keys in FALCON via single-trace analysis. *Journal of Cryptographic Engineering*, 16(3):14, Aug 2026.

[QGQ+25] Yuheng Qian, Jing Gao, Yuhan Qian, Yaoling Ding, and An Wang. STAR: Self-training assisted refinement for side-channel analysis on cryptosystems. *Cryptography*, 9(4), 2025.

[QLZ+23] Zehua Qiao, Yuejun Liu, Yongbin Zhou, Mingyao Shao, and Shuo Sun. When NTT meets SIS: Efficient side-channel attacks on Dilithium and Kyber. Cryptology ePrint Archive, Paper 2023/1866, 2023.

[SE94] C. P. Schnorr and M. Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Mathematical Programming*, 66(2):181–199, September 1994.

[SKL+20] Bo-Yeon Sim, Jihoon Kwon, Joohee Lee, Il-Ju Kim, Tae-Ho Lee, Jaeseung Han, Hyojin Yoon, Jihoon Cho, and Dong-Guk Han. Single-trace attacks on message encoding in lattice-based KEMs. *IEEE Access*, 8:183175–183191, 2020.

[SMY09] François-Xavier Standaert, Tal G. Malkin, and Moti Yung. A unified framework for the analysis of side-channel key recovery attacks. In Antoine Joux, editor, *Advances in Cryptology - EUROCRYPT 2009*, pages 443–461, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[Ste89] Jacques Stern. A method for finding codewords of small weight. In Gérard Cohen and Jacques Wolfmann, editors, *Coding Theory and Applications*, pages 106–113, Berlin, Heidelberg, 1989. Springer Berlin Heidelberg.

[STM26] STMicroelectronics. *STM32F415xx STM32F417xx Datasheet*, 9 2026. Rev. 10. Available: https://www.st.com/resource/en/datasheet/stm32f415rg.pdf.

[TOS25] Tolun Tosun, Elisabeth Oswald, and Erkay Savaş. Non-profiled higher-order side-channel attacks against lattice-based post-quantum cryptography. *IACR Communications in Cryptology*, 2(3), 2025.

[TS24] Tolun Tosun and Erkay Savaş. 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, January 2024.

[VCMKS12] Nicolas Veyrat-Charvillon, Marcel Medwed, Stéphanie Kerckhof, and François-Xavier Standaert. Shuffling against side-channel attacks: A comprehensive study with cautionary note. In Xiaoyun Wang and Kazue Sako, editors, *Advances in cryptology – ASIACRYPT 2012*, page 740–757, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[WNGD24] Ruize Wang, Kalle Ngo, Joel Gärtner, and Elena Dubrova. Unpacking needs protection: A single-trace secret key recovery attack on Dilithium. *IACR Communications in Cryptology*, 1(3), 2024.

[ZHC+26] Yuhan Zhao, Dalin He, Wei Cheng, Yuejun Liu, Jingdian Ming, and Yongbin Zhou. When module lattice leaks: Horizontal fusion attacks on ML-DSA implementation. Cryptology ePrint Archive, Paper 2026/1904, 2026.