Single Trace is All It Takes: Efficient Side-channel Attack on Dilithium

Zehua Qiao[1,2,4], Yuejun Liu[3], Yongbin Zhou[1,3], Yuhan Zhao[3] and Shuyi Chen[3]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{qiaozehua}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3] School of Cyber Science and Engineering, Nanjing University of Science and Technology, Nanjing, China liuyuejun@njust.edu.cn
[4] Key Laboratory of Cyberspace Security Defense

Abstract.
As we enter 2024, the post-quantum cryptographic algorithm Dilithium, which emerged from the National Institute of Standards and Technology post-quantum cryptography competition, has now reached the deployment stage. This paper focuses on the practical security of Dilithium. We performed practical attacks on Dilithium2 on an STM32F4 platform. Our results indicate that an attack can be executed with just two signatures within five minutes, with a single signature offering a 60% probability of recovering the private key within one hour. Specifically, we analyze the polynomial addition $z = y + \mathbf{cs}_1$. The attack is conducted in two phases: initially applying side-channel analysis to recover the values of y or $\mathbf{cs}_1$, followed by solving an equation system of $\mathbf{cs}_1$ with error. We introduce using Linear Regression-based profiled attack to recover y, leveraging the mathematical properties of adding large and small numbers, requiring only one trace to achieve a 40% success rate. In contrast, a CNN-based template attack, trained with leakage from 200 signatures, enables $\mathbf{cs}_1$ recovery from a single trace with a 74% success rate. Further, by exploiting the constraint $z = y + \mathbf{cs}_1$, the combined leakages of y and $\mathbf{cs}_1$ increase the success rate for $\mathbf{cs}_1$ recovery to 92%. Additionally, we propose a constrained optimization-based residual analysis to solve the equation set $\mathbf{cs}_1 = b$ with error. This method can function independently or as a preprocessing step in combination with Belief Propagation or Integer Linear Programming. Experimental results show that with a 95% correctness rate in the equation set, this method can directly recover the private key $\mathbf{s}_1$ with an 83% success rate in just five seconds. Even with a correctness rate as low as 5%, the method can still recover the private key $\mathbf{s}_1$ in 5 minutes using the system of equations generated by about 200 signatures.

Keywords: Lattice-based Cryptography · CNN · Side-channel Attacks · Dilithium

1 Introduction

The rapid development of quantum computing poses a significant threat to public-key cryptography based on the computational complexity of problems like large integer factorization and discrete logarithm problems. Once a general-purpose quantum computer is successfully developed, the use of Shor's algorithm, proposed in 1994 [Sho94], will enable these algorithms to be broken in polynomial time. Consequently, the National Institute of Standards and Technology (NIST) initiated the global standardization competition for Post-Quantum Cryptography (PQC) and designated CRYSTALS-Dilithium

(abbr.Dilithium) as the digital signature standard, releasing a draft standard in 2023.

Dilithium [DKL+18] is a digital signature scheme founded on the hardness of lattice problems, constructed using the Fiat-Shamir paradigm over the polynomial ring $\mathbb{Z}_{q[x]}/(x^n+1)$. Due to its excellent performance in terms of operational efficiency and theoretical security, Dilithium is recommended by NIST as the prime scheme for digital signature.

Although post-quantum cryptographic algorithms theoretically possess the capability to resist attacks from both classical computers and general-purpose quantum computers, in practice, their implementation and deployment inevitably face real-world physical attacks, among which side-channel analysis is the most typical and widespread. This attack method leverages leakages from cryptographic operations, such as power consumption [KJJ99], electromagnetic emissions [QS01], and execution timing [Koc96], to extract sensitive information. Over nearly three decades, side-channel analysis has achieved numerous breakthroughs in the field of cryptanalysis, successfully conducting practical analyses against cryptographic algorithms such as AES [BCO04], RSA [BJL+14], and ECC [Ols04].

Since NIST initiated the post-quantum cryptography selection process, several studies have conducted practical analyses on Dilithium. One approach involves direct attacks on operations involving the private key. For private keys in the normal domain, Han *et al.* [HLK+21] implemented practical attacks on the Number Theoretic Transform (NTT) operations of the Dilithium private key using machine learning-based template attacks. Wang *et al.* [WNGD23] introduced an attack targeting the private key unpacking phase of the signing process, achieving a 9% probability of recovering the private key with a single signature, and a 100% success rate using 74 signatures when assuming $\mathbf{t}_0$ is known. For private keys in the NTT domain, Chen *et al.* [CKA+21, FDK20] recovered the private key using Correlation Power Analysis (CPA) and improved the speed of attacks by integrating a divide-and-conquer strategy. Furthermore, Qiao *et al.* [QLZ+23b] designed a fast attack method based on CPA, and used the LLL algorithm to solve for erroneous NTT domain private keys, achieving full Dilithium private key recovery within one minute. Additionally, Tosun *et al.* [TS24] proposed a zero-value filtering attack that quickly attacked the polynomial multiplication in incomplete NTT implementations.

Another common analysis approach first involved obtaining non-key sensitive intermediate values, followed by using mathematical analysis to recover the private key. The polynomial addition operation $z = y + \mathbf{cs}_1$ used in the signing process was an important target for analysis. Liu *et al.* [LZS+21, QLZ+23a] introduced a strategy for random leakage attacks targeting y. They used a public template attack to recover the lower bits of y, leveraging these bits to reduce the private key recovery problem to an integer Learning With Errors (LWE) problem solvable in polynomial time. Marzougui *et al.* [MUTS22] profiled all possible values of y, identifying special values such as 0 during the attack, and then used integer linear programming (ILP) to solve the erroneous equations for $\mathbf{cs}_1$ and recover the private key $\mathbf{s}_1$. Additionally, Berzati *et al.* [BVC+23] attacking smaller sensitive intermediate values like $\mathbf{w}_0$, using the Central Limit Theorem to design a filtering algorithm that more accurately identified special values $w_0 = 0$. Ultimately, they employed the Majority Vote method to eliminate the impact of incorrect results.

The high-dimensional properties of lattice-based cryptography typically prevents side-channel attacks from recovering the entire private key with 100% success. Therefore, existing approaches are generally divided into two phases: initially, side-channel attacks aim to recover as much information as possible about the private key and other sensitive variables with the highest feasible success rate; subsequently, mathematical analysis methods are employed to reconstruct the complete private key. In the case of Dilithium, when directly attacking variables such as y or $\mathbf{w}_0$, their broad range of individual coefficient values often leads to the adoption of a strategy that filters for special values to enhance the success rate of side-channel attacks. However, this approach requires a substantial

Parts of this work was submitted to Fincryptography at 15:00pm Mar 30, 2024 AOE.

number of traces for profiling and the attack process. Moreover, these sensitive variables are generated randomly during each signing process, necessitating the use of only one trace for the side-channel attack, which complicates achieving a high success rate.

In the mathematical analysis phase, when the success rate of side-channel attacks is low, employing methods like ILP and majority voting to solve the erroneous linear equations involving $\mathbf{cs}_1$ can require significant time and may sometimes be impractical. Innovatively, Bronchain *et al.* [BAE+23] proposed the use of the Belief Propagation (BP) algorithm to solve these erroneous equations for $\mathbf{cs}_1$ and conducted simulated experiments on both fault injection and side-channel attacks targeting y and $\mathbf{cs}_1$. Under the Hamming model with a Signal-to-Noise Ratio (SNR) of 100, it was possible to recover the private key $\mathbf{s}_1$ using just four signatures. However, when the success rates of side-channel or fault injection attacks are low, this method necessitates a large number of equations. As a result, the graph corresponding to the BP algorithm would be substantial, and the iterative process to reach stability could increase the likelihood of computational overflow, presenting challenges in terms of both time and spatial overhead.

In this paper, we explore side-channel attack methods for Dilithium that aim to recover the system with the fewest possible signatures while achieving a higher success rate. We also examine mathematical methods capable of solving linear equation systems with high error rates. Our specific contributions are as follows:

- In side-channel attacks targeting y, we exploit the operation $z = y + \mathbf{cs}_1$, which involves adding a large number to a smaller one, to recover the high bits of y. We propose a linear regression-based attack that eliminates noise from the already determined high bits, thereby not only increasing the attack's success rate but also enhancing its efficiency. Experimental results show that with just a single trace, we achieve a 40% success rate in recovering any y.

- In attacks targeting $\mathbf{cs}_1$, the relatively narrow range of possible values allows attackers to profile and launch attacks on each specific value. We propose a side-channel approach using a CNN model that, compared to traditional template attack methods, boosts the success rate from 57% to 74%.

- To further enhance the success rate of side-channel attacks, we propose a method that utilizes attacks on both y and $\mathbf{cs}_1$ simultaneously. We leverage the constraint $z = y + \mathbf{cs}_1$ to merge the results, thereby improving the success rate of recovering $\mathbf{cs}_1$. Experimental results demonstrate that this method increases the recovery success rate of $\mathbf{cs}_1$ from a maximum of 74% to 92%.

- We develop a constrained optimization-based residual analysis to rapidly resolve erroneous integer linear equations for $\mathbf{cs}_1$. Experimental results show that with a 95% side-channel success rate for $\mathbf{cs}_1$, using just one signature can recover $\mathbf{s}_1$ with an 80% success rate in five seconds. Even with only 5% of the equations being correct, this method can still recover $\mathbf{s}_1$ in less than five minutes using equations corresponding to 220 signatures. Additionally, this method can serve as a preliminary step for other techniques such as BP and ILP, especially effective when the system of equations contains a large number of errors, as it helps eliminate a large number of incorrect equations to improve analytical efficiency.

- We conduct practical attacks on Dilithium2's reference implementation using an STM32F4, targeting three scenarios: leaks from y, $\mathbf{cs}_1$ alone, and from both y and $\mathbf{cs}_1$ simultaneously. Experimental results show that in all three scenarios, the private key is successfully recovered within three minutes, requiring 6, 3, and 2 signatures respectively. By leveraging leaks from both y and $\mathbf{cs}_1$ and using just one signature, the constrained optimization-based residual analysis method alone manages

to recover the private key $\mathbf{s}_1$ with a 20% success rate. Furthermore, Furthermore, by incorporating integer linear programming and limiting the attack duration to one hour, the success rate for recovering $\mathbf{s}_1$ increases to 60%.

2 Preliminaries

2.1 Dilithium

Dilithium is a digital signature scheme based on the Module Learning with Errors (MLWE) and Module Short Integer Solution (MSIS) problems. Dilithium offers different security levels to meet the security and performance requirements of various devices, making it suitable for a wide range of uses. Tab.1 shows the parameters for each security level.

Table 1: Dilithium parameters at different NIST security levels

NIST Security Level	2	3	5
d [dropped bits from $\mathbf{t}$]	13		
τ[# of non-zero coefficients in $\mathbf{c}$]	39	49	60
γ_1 [cofficient range of $\mathbf{y}$]	131,072	524,288	
γ_2 [low-order rounding range]	95,232	261,888	
$(m \times n)$ [dimensions of $\mathbf{A}$]	(4,4)	(6,5)	(8,7)
η [private key range]	2	4	2
β $[\tau \cdot \eta]$	78	196	120

Dilithium operates within the cyclotomic ring $\mathbb{R}_q^n$, where each coefficient is defined in the finite field $\mathbb{Z}_q$. The constants $q = 8380417$ and $n = 256$ are fixed across all security levels, ensuring a uniform foundation for operations. The algorithm consists of three basic processes: key generation, signing, and signature verification. Our work primarily focuses on the signing process.

Algorithm 1 Dilithium Sign(sk,M)

Input: $sk = (\rho, K, tr, \mathbf{s}_1, \mathbf{s}_2, \mathbf{t}_0), M$
Output: *signature*
1: $\mathbf{A} \in \mathbb{R}_q^{m \times n} := \text{ExpandA}(\rho)$
2: $\mu \in \{0,1\}^{384} := \text{CRH}(tr||M)$
3: $\kappa := 0, (\mathbf{z}, \mathbf{h}) := \perp$
4: $\rho' \in \{0,1\}^{384} := \text{CRH}(K||\mu)$ (or $\rho' \leftarrow \{0,1\}^{384}$)
5: $\hat{\mathbf{A}} = \text{NTT}(\mathbf{A}), \hat{\mathbf{s}}_1 = \text{NTT}(\mathbf{s}_1)$
6: $\mathbf{y} \in S^n_{\gamma_1 - 1} := \text{ExpandMask}(\rho', \kappa)$
7: $\mathbf{w} := \text{NTT}^{-1}(\hat{\mathbf{A}} \circ \text{NTT}(\mathbf{y}))$
8: $\mathbf{w}_1 := \text{HighBits}_q(\mathbf{w}, 2\gamma_2)$
9: $\tilde{\mathbf{c}} \in \{0,1\}^{256} := \mathbf{H}(\mu||\mathbf{w}_1)$
10: $\hat{\mathbf{c}} := \text{NTT}(\text{SampleInBall}(\tilde{\mathbf{c}}))$
11: $\mathbf{z} := \mathbf{y} + \text{NTT}^{-1}(\hat{\mathbf{c}} \circ \hat{\mathbf{s}}_1)$
12: $\mathbf{r}_0 := \text{LowBits}_q(\mathbf{w} - \mathbf{cs}_2, 2\gamma_2)$
13: **if** $||\mathbf{z}||_\infty \geqslant \gamma_1 - \beta$ **or** $||\mathbf{r}_0||_\infty \geqslant \gamma_2 - \beta$
then $\kappa := \kappa + l$, goto 6
14: **else**
15: $\mathbf{h} := \text{MakeHint}_q(-\mathbf{ct}_0, \mathbf{w} - \mathbf{cs}_2 + \mathbf{ct}_0, 2\gamma_2)$
16: **if** $||\mathbf{ct}_0||_\infty \geqslant \gamma_2$ **or** the # of $1^{'}$s in $\mathbf{h}$ is greater than ω
then $\kappa := \kappa + l$, goto 6
17: **return** *signature* $= (\mathbf{z}, \mathbf{h}, \tilde{\mathbf{c}})$

The signing process of Dilithium, as outlined in Alg.1, commences with the input of private key sk and message M. Initially, the algorithm expands the private key ρ to construct a structured matrix $\mathbf{A}$ within $\mathbb{R}_q^{m\times n}$ using the ExpandA function, followed by generating a 384-bit string μ from tr and message M through the Collision Resistant Hash (CRH) function. It initializes κ and sets $(\mathbf{z}, \mathbf{h})$ to null, then creates a 384-bit string ρ either by hashing κ concatenated with μ or by selecting randomly depend on whether the algorithm is deterministic or probabilistic. Subsequently, NTT is applied to both $\mathbf{A}$ and the private key $\mathbf{s}_1$, generating $\hat{\mathbf{A}}$ and $\hat{\mathbf{s}}_1$. A masking vector $\mathbf{y}$ is derived from ρ' and κ, within the set S_{γ_1-1}. The algorithm computes $\mathbf{w}$ by multiplying $\hat{\mathbf{A}}$ with the NTT domain $\hat{\mathbf{y}}$ and applying Inverse NTT (INTT), then extracts high-order bits from $\mathbf{w}$ to generate a challenge vector $\tilde{\mathbf{c}}$, which is a part of the signature. The algorithm employs a rejection sampling mechanism to ensure the generated vectors $\mathbf{z}$ and $\mathbf{r}_0$ to meet specific security criteria. Otherwise, the algorithm regenerate $\mathbf{y}$ and repeat the above process until a valid signature comprising a hint vector $\mathbf{h}$, $\tilde{\mathbf{c}}$ and $\mathbf{z}$ is generated.

2.2 Linear Regression-based Profiled Attacks

Schindler *et al.* [SLP05] introduced Linear Regression-based (LR) profiled attacks. Compared to the Hamming Weight (HW) leakage model in traditional template attacks [CRR02], they take into account that different bits might have different leakage weights, and this nuance can be accurately identify using linear regression, providing a more precise characterization of practical leakage scenarios. This marks a significant shift in the leakage model.

The power consumption model established using linear regression is formulated as:

$$m(y) = \sum_{i=0}^{l_y} a_i \rho(y_i) + a_{l_y+1} \tag{1}$$

Here, y represents the targeted data, y_i denotes the i-th bit from low to high, and l_y denotes the bit length of y. The coefficients a_i indicate the leakage weight of each bit, while $\rho(y_i)$ is a mapping function that ensures the model includes the influence of $y_i = 0$, which is expressed as follows:

$$\rho(y_i) = \begin{cases} 1 & \text{if } y_i = 1 \\ -1 & \text{if } y_i = 0 \end{cases} \tag{2}$$

In the phase of building templates, real power leakages L are used to calculate the coefficients a_i via linear least squares. The model $m(y)$ subsequently serves as the mean in traditional template attacks, requiring an additional step to compute the covariance matrix Σ for template generation. During the attack phase, the probability density function for any captured leakage L is outlined as:

$$f[L|Y=y] = \frac{1}{\sqrt{(2\pi)^k|\Sigma|}} \exp\left(-\frac{1}{2}(L-m(y))^T\Sigma^{-1}(L-m(y))\right) \tag{3}$$

where k is the dimension of L. Utilizing Bayes' theorem, this is reformulated into the desired probability $f[Y=y|L]$, illustrated as:

$$f[Y=y|L] = \frac{f(L|y)p(y)}{\sum_{y'} f(L|y')p(y')} \tag{4}$$

Assuming equal probabilities for each y, the expression can be simplified to:

$$f[Y=y|L] = \frac{f(L|y)}{\sum_{y'} f(L|y')} \tag{5}$$

2.3 CNN-based Template Attacks

In the rapidly evolving domain of side-channel attacks, the adoption of deep learning approaches, particularly CNNs, has demonstrated excellent technical effectiveness [MPP16, ZBHV20, WAGP20]. The deployment of CNN in side-channel attacks is predicated on the assembly of a comprehensive training dataset, comprising leakage traces, plaintext-ciphertext pairings, and corresponding cryptographic keys. Each trace is labeled with sensitive intermediate values, thus categorizing the data for the training phase of the CNN.

The effectiveness of CNNs is attributed to their hierarchical architecture that begins with convolutional layers responsible for initial feature extraction. This is followed by pooling layers that reduce the feature set's dimensionality, and fully connected layers that undertake the task of classification. Convolutional layers employ a set of filters—each with unique weights and biases—to conduct convolution operations on the input data. This process effectively captures and highlights essential patterns. Pooling layers simplify the feature set by summarizing data within specific input regions, applying max and average pooling techniques to preserve vital information efficiently. Fully connected layers integrate these refined features to produce the final output classifications. The strategic placement of batch normalization layers between select convolutional and pooling stages significantly boosts the network's efficiency and stability during training by standardizing the inputs to each layer. Its mathematical formulation can be succinctly represented as [ZBHV20]:

$$g(x) = f \circ [\lambda]^{n_1} \circ [\delta \circ [\alpha \circ \gamma]^{n_2}]^{n_3} . \tag{6}$$

Here γ, α, δ, λ, and f represent convolutional layers, activation functions, pooling layers, fully connected layers, and the activation function of the output layer, respectively. The variables n_1, n_2, and n_3 indicate the respective counts of these computational components, illustrating the CNN's structural depth and complexity.

The training phase is crucial for the CNN , equipping the model with the ability to accurately identify patterns within leakage traces. Once trained, the CNN can effectively predict sensitive intermediate values from previously unseen traces and recover sensitive information such as private keys.

3 Side-channel Attacks Against Dilithium

According to the polynomial addition operation $z = y + \mathbf{cs}_1$ in the signing algorithm, the known signature z offers an indirect pathway to deduce $\mathbf{cs}_1$ by initially recovering y using side-channel attacks. Alternatively, $\mathbf{cs}_1$ can be directly recovered via side-channel attacks. By merging leakages of y and $\mathbf{cs}_1$, we expect a significant enhancement in the analysis results. The methodologies for conducting side-channel attacks on both y and $\mathbf{cs}_1$ will be detailed in this section.

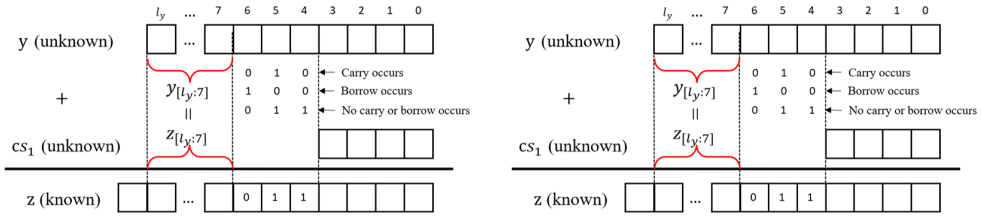
3.1 Attacking y with LR-Based Side-channel Attack

For the mask polynomial $\mathbf{y}$, each coefficient operating within the range of $q = 8380417$. Considering that $\mathbf{y}$ is randomly generated in each signature, we have to recover $\mathbf{y}$ utilizing only one trace. Conventional template attacks using the identity model are theoretically feasible but challenging in practice, due to the high overhead of accurate templates building and the low success rate of template matching.

For the polynomial addition $z = y + \mathbf{cs}_1$, where $\mathbf{cs}_1$ ranges within $(-\beta, \beta)$ and follows a Gaussian distribution, the value of β depends on the chosen security level, as outlined in Tab.1. Notably, for Dilithium3, the maximum value of β remains below 198. This fact makes it unnecessary to enumerate all possible y values when executing a template

attack aimed at recovering y. Given the known signature z, it is clear that viable y values, which are relevant to the trace targeted in the attack, are effectively limited to the interval $(z - ||\mathbf{cs}_1||_\infty, z + ||\mathbf{cs}_1||_\infty)$. This restriction significantly accelerates the attack.

Liu *et al.* [LZS+21, QLZ+23a] have highlighted the methodology for inferring the high-bit information of y when z is available, especially when $\mathbf{cs}_1$ is significantly smaller than y. In the Dilithium signature process, both the signature z and the random number y can take on one of 2^{18} possible values, whereas the intermediate value $\mathbf{cs}_1$ is typically much smaller than both z and y. This disparity allows for the deduction of partial information about the random number y by exploiting the arithmetic operation of adding a larger number to a smaller one. Fig.1 illustrates this principle. Assuming $||\mathbf{cs}_1||_\infty < 2^4$, and when $z_{[i-1:\tau]} = 10\ldots00_2$ or $z_{[i-1:\tau]} = 01\ldots11_2$, we identify three scenarios enabling the attacker to derive a segment of y (specifically $y_{[l_y:i]}$) through its addition with $\mathbf{cs}1$ to result in $z_{[i-1=6:\tau=4]} = 100_2$ or 011_2. These scenarios include instances of addition that involve carrying, borrowing, or neither. Crucially, none of these instances affect y_7 or higher bits, leading to the inference that $y_{[l_y:i]} = z_{[l_y:i]}$. This analysis demonstrates that in many instances, the high-bit information leakage of y can largely be ignored during a template attack, significantly enhancing the attack's success rate.



(a) $z_{[i-1=6:\tau=4]} = 100_2$ (b) $z_{[i-1=6:\tau=4]} = 011_2$

Figure 1: Example of $y_{[l_y:i]} = z_{[l_y:i]} (i = 7, \tau = 4)$

LR-based profiled attacks offer a more simplified method for creating models, necessitating fewer training samples to approximate the practical leakage corresponding to the complete target value. This method enhances template precision by delineating leakage features for every bit of the target value. In the template matching phase, the methodology involves iterating over all conceivable y' values, comparing the theoretical leakage predicted by the LR model against practical captured trace. Crucially, bits within varying y' values that stay unchanged don't affect the matching process, rendering this strategy particularly suited for instances involving polynomial addition $z = y + \mathbf{cs}_1$ in the context of the Dilithium cryptographic algorithm.

The detailed steps for conducting an attack on y are as follows:

(1) Templates building: Utilize the captured traces along with their respective labels to identify the leakage for each bit of y. Subsequently, a LR model is then build to represent this leakage data accurately.

(2) Traces capturing: Capture traces associated to y from the target device.

(3) Template Matching: For each captured trace, iterate over all possible $\mathbf{cs}_1$ values. By combining with the known signature z, deduce all possible y values. The exact y value is determined by comparing the theoretical leakage, as predicted by the LR model, with the practical captured leakage, thus identifying the precise match through template matching.

The Gaussian-like distribution of $\mathbf{cs}_1$ plays a crucial role in enhancing the success rate of attacks by allowing for the exclusion of less probable cases. This approach is effective across different security levels of Dilithium, where the range of $\mathbf{cs}_1$ is detailed in Tab.2.

Using Dilithium2 as an example, we find that about 95% of the data falls within $||\mathbf{cs}_1||_\infty < 2^4$, with the remainder 5%, as detailed in Tab.2, covering a broader value range

Table 2: Probability of $||\mathbf{cs_1}||_\infty < 2^\tau$ in Dilithium.

Security Level	τ=3	τ=4	τ=5	τ=6	τ=7
Dilithium2	0.57	0.95	0.99	1	1
Dilithium3	0.36	0.64	0.93	0.99	1
Dilithium5	0.65	0.86	0.99	1	1

of $(\pm 16, \pm 78)$. These candidates encompass a significantly larger value space but occur with much less frequency. Including all these potential values in the template match might decrease the success rate due to the dilution of focus on the most possible candidates. By excluding these less likely candidates from the analysis, the success rate and efficiency of the attack can be significantly enhanced.

3.2 Attacking $\mathbf{cs_1}$ with CNN-Based Side-channel Attack

In addition to focusing on the random number y, directly targeting the $\mathbf{cs}_1$ emerges as a viable alternative. Due to its smaller value space, $\mathbf{cs}_1$ naturally presents more advantageous conditions for side-channel attacks. The attack scenario is similar to y in that $\mathbf{cs}_1$ can only be recovered from a single trace. This highlights the crucial importance of template quality of the attack.

Lattice-based cryptographic schemes, such as Dilithium, are characterized by a high-dimensional environment filled with numerous repetitive and independent operations throughout the polynomial processing stages. Specifically, for Dilithium2, the creation of a valid signature requires, on average, four instances of rejection sampling. This translates to approximately $256 \times 4 \times 5$ relevant operations for $\mathbf{cs}_1$. When accounting for $\mathbf{cs}_2$, a single legitimate signature, in a deterministic implementation, can produce an average of 10,240 samples. Even in randomised implementations, it's possible to gather a substantial 2048 samples. This wealth of data is particularly suitable for the demands of deep learning, which requires a large dataset for pre-training in order to effectively develop a distinguisher.

During the model selection phase, it is crucial to consider that numerous samples from each signature are produced at different times. This necessitates alignment operations to pinpoint leakage features, which inevitably generate offsets. Deep learning approaches, particularly CNN, have demonstrated impressive capabilities in handling data marked by such offsets. This efficiency largely stems from the CNN architecture, which integrates convolutional layers and pooling layers. These layers are tailored to extract and refine features from data, even in cases of alignment variability. As a result, CNN-based methods for side-channel analysis are distinguished by their ability to conduct effective feature extraction from datasets with inherent offsets, showcasing their robustness in extracting relevant information for analysis.

In our research, we adopt a CNN model inspired by the methodology of Zaid *et al.* [ZBHV20] as described in their study, which utilizes a three-layer convolutional setup. This model is particularly effective in processing complex side-channel datasets like ASCAD, which includes countermeasures such as random delay and first-order masking. It achieves a Guessing Entropy (GE) of 1 with a relatively small number of traces—244 for offset $N = 50$ and 270 for offset $N = 100$.

Our implementation utilizes a comparable three-layer CNN architecture. This structure is composed of convolutional layers interspersed with pooling and batch normalization operations. The design facilitates the gradual recognition of features, ranging from simple to intricate, and culminates in a dense layer equipped with a softmax activation function for precise classification of the processed inputs. Details of our CNN architecture are delineated in Tab.3 below.

By adopting convolutional approach, our model benefits from the ability to capture and learn complex patterns in the side-channel traces effectively. The architecture's progressive feature extraction and condensation stages allow for efficient representation of the relevant

Table 3: CNN Hyperparameters

Hyperparameter	**Configuration**
Optimizer	`Adam`
Convolution Layers	3
Convolution Filters	[12, 24, 48]
Convolution Kernel	[64, 128, 256]
Convolution Stride	[1, 6, 1]
Pooling Type	`avgPooling`
Pooling Size	[2, 4, 4]
Pooling Stride	[2, 4, 4]
Batch Normalization	After each pooling
Dense Layers	1
Neurons	Number of classes (variable)
Activation Function	`ReLU`
Learning Rate	0.0004
Batch-Size	1600
Epochs	150
Loss Function	`categorical_crossentropy`

information, enabling robust classification performance while maintaining computational efficiency. Certainly, the optimal model hyperparameters will be varied for different data and can be adjusted by random or grid search.

3.3 Enhancing Success Rates through Integrated Results of y and $\mathbf{cs}_1$

In side-channel analysis, directly recovering the complete value of a target on the ARM platform from a single trace presents notable difficulties. Nevertheless, it's more practical to determine the HW of variables such as y and $\mathbf{cs}_1$. Crucially, having exclusive access to the HW information of y and $\mathbf{cs}_1$, alongside the signature $z = y + \mathbf{cs}_1$, enables the deduction of $\mathbf{cs}_1$ under certain conditions. Merging the outcomes from attacks that target these essential intermediate values can significantly improve the success rate of the analyses. Bronchain *et al.* [BAE+23] applied this concept in simulation experiments within the HW model for $\mathbf{cs}_1$, though this approach may lead to scenarios with multiple potential candidate values.

$$\begin{aligned} &P(Y = y_i|L(y)) \qquad y_i \in (z-\beta, z+\beta) \\ &P(X = x_i|L(x)) \qquad x_i \in (-\beta, \beta) \\ &P(X = x_i|L) = P(X = z - y_i|L(y)) \times P(X = x_i|L(x)) \end{aligned} \tag{7}$$

Our side-channel attacks aim to directly recover the full value of the target, rather than its HW. Typically, even if the attack does not recover the correct result, the probability of other candidate values in the probability distribution vector that have the same HW as the correct value will be comparably higher. Assuming $\mathbf{cs}_1 = x$, Eq.7 detailing our computational procedure. In practical analysis, particularly when examining targets y and $\mathbf{cs}_1$, we preprocess by normalizing the set of probability vectors derived from the side-channel analysis before computing the final probability. This strategy leads to a uniquely deterministic solution, theoretically surpassing the effectiveness of focusing on a single target in isolation.

In contrast to traditional cryptographic algorithms like ECC and RSA, Dilithium features a more complex computational framework. This complexity introduces a variety of sensitive values during the computation, which can be leveraged through side-channel analysis, thus amplifying the security risks. By consolidating attack outcomes across

different sensitive values, it becomes possible to break the algorithm's security with minimal expenditure, possibly even leading to the recovery of the private key.

4 Solving Equations with Errors in $\mathbf{cs}_1$ for Dilithium

After extracting $\mathbf{cs}_1$ via side-channel or fault attacks, next we need to solve error-containing integer linear equations of $\mathbf{s}_1$ using the known challenge ciphertext $\mathbf{c}$. Achieving 100% success rate in determining $\mathbf{cs}_1$ is inherently challenging, regardless of the attack method. Traditionally, the Big-M method has been used to convert these challenges into ILP problem. However, this method's efficiency drops as errors increase. To counter this, we propose a constrained optimization-based residual analysis to efficiently solve error-containing integer linear equations in Dilithium.

4.1 Constrained Optimization-Based Residual Analysis

This section delineates the computational methodology for deriving $\mathbf{cs}_1$ from the given challenge vector $\mathbf{c}$, which is a 256-dimensional vector consisting mainly of zeros, and includes -1, 0, and 1 as its elements. The computation of $\mathbf{c} = (c_0, c_1, \ldots, c_{255})$ into the coefficient matrix $\mathbf{C}$ is achieved through cyclotomic transformations, executed within the confines of a finite field. The detailed calculations are as follows:

$$\begin{bmatrix} c_0 & -c_{n-1} & -c_{n-2} & \cdots & -c_2 & -c_1 \\ c_1 & c_0 & -c_{n-1} & \cdots & -c_3 & -c_2 \\ c_2 & c_1 & c_0 & \cdots & -c_4 & -c_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c_{n-2} & c_{n-3} & c_{n-4} & \cdots & c_0 & -c_{n-1} \\ c_{n-1} & c_{n-2} & c_{n-3} & \cdots & c_1 & c_0 \end{bmatrix} \begin{bmatrix} s_0 \\ s_1 \\ \vdots \\ s_{n-2} \\ s_{n-1} \end{bmatrix} = \mathbf{Cs}$$

The special values of $\mathbf{c}$ and $\mathbf{s}$ allow the entire computation to be performed without modular operations. This characteristic permits the use of methods within the normal domain to address the problem effectively.

Consider the linear system $\mathbf{As} = \mathbf{b}$, with $\mathbf{A}$ as an $m \times n$ matrix derived from the $\mathbf{c}$ generated by multiple signatures, $\mathbf{s}$ as an n-vector representing the unknown integer-valued private keys ($\mathbf{s}_1$ or $\mathbf{s}_2$), and $\mathbf{b}$ as an m-dimensional vector obtained from practical attack. Assuming an attacker achieves a 30% success rate in acquiring $\mathbf{b}$, the collection of 10 signatures yields 2,560 equations with 768 being correct. The goal is to identify an integer solution for $\mathbf{s} \in \{-\eta, \ldots, \eta\}^n$ that maximizes the count of accurately fulfilled equations. Direct optimization of this count is complex. Nevertheless, if 256 correct equations can be identified from the system of equations, the private key can be recovered next by solving the simplified equation set. This scenario translates into a continuous optimization problem, focusing on minimizing the residuals between predicted and observed values.

The process of eliminating erroneous equations is divided into two phases. The initial phase involves transforming the problem into a continuous optimization framework. The attacker aims to minimize the sum of squared residuals across all equations. This objective is mathematically expressed as:

$$\min_{\mathbf{s}} \frac{1}{2} \|\mathbf{As} - \mathbf{b}\|_2^2 \quad \text{s.t.} -\eta \leq \mathbf{s}_i \leq \eta \tag{8}$$

The initial phase of filtering erroneous equations can be reformulated as a large-scale bound-constrained minimization challenge. In tackling such problems, Branch *et al.* [BCL99] have introduced a technique by adapting the Coleman-Li trust region and interior method. For computational efficiency, this technique may employ sparse Cholesky factorization or the conjugate gradient method.

Following the derivation of an approximate solution, denoted as $\mathbf{s}^*$, from Eq.8, the subsequent phase involves evaluating the residuals for each equation in the system. This evaluation aims to measure how well each equation aligns with the obtained solution. The residual for the i-th equation is calculated as $\mathbf{r}_i = |\mathbf{A}_i\mathbf{s}^* - \mathbf{b}_i|$, where $\mathbf{A}_i$ represents the i-th row of $\mathbf{A}$, and $\mathbf{b}_i$ is the i-th element of $\mathbf{b}$. Equations with the highest residuals are presumed to be inaccurate and are identified as candidates for removal in further iterations. The principle behind this exclusion process is that are most likely to be wrong from the current solution can lead to a more accurate approximation of the true solution by reducing the influence of potential errors.

The iteration process continues next, recomputing the solution $\mathbf{s}^{(k)}$ using the updated equation set $\mathbf{A}^{(k)}$ and $\mathbf{b}^{(k)}$ that omit previously identified incorrect equations at each k-th iteration. The mathematical formulation is as follows:

$$\mathbf{s}^{(k)} = \min_{\mathbf{s}} \frac{1}{2} \left\| \mathbf{A}^{(k)}\mathbf{s} - \mathbf{b}^{(k)} \right\|_2^2 \quad \text{s.t.} -\eta \leq \mathbf{s}_i \leq \eta \tag{9}$$

Iteration proceeds until the solution reaches a satisfactory level of accuracy, marked by minimal residual differences between the estimated solution and the actual data represented by $\mathbf{b}^{(k)}$ in the updated equation set.

In order to get the final result, the continuous solution $\mathbf{s}^*$ evaluated for its closeness to the nearest integers within the bounds $\{-\eta, \ldots, \eta\}^n$. The conversion to an integer solution, $\mathbf{s}_{\text{int}}$,involves rounding each element of $\mathbf{s}^*$ to its closest integer value:

$$\mathbf{s}_{\text{int},i} = \text{round}(\mathbf{s}_i^*), \quad \forall i = 0, \ldots, n-1 \tag{10}$$

This step assumes the proximity of the continuous solution to the actual, integer-valued solution allows for effective rounding, achieving a solution that fulfills most, if not all, equations form the filter set. Subsequent verification of the integer solution is advised to confirm its adequacy in satisfying the linear system to an acceptable degree.

Algorithm 2 Constrained Optimization-Based Residual Analysis

Input: $\mathbf{A}, \boldsymbol{sca_r}, bounds, e_{th}, d_{num}, r_{num}, s_{num}$
Output: $\mathbf{s}$

```
available_ind ← InitializeIndices(A)
err_weights ← zeros(|A|)
while |available_ind| > r_num do
    s_ind ← RandomSample(available_ind, s_num)
    s ← SolveLSQ(A[s_ind], sca_r[s_ind], bounds)
    residuals ← CalculateResiduals(A[s_ind], sca_r[s_ind], s)
    err_weights ← UpdateWeights(residuals, err_weights, d_num)
    available_ind ← UpdateIndices(err_weights, e_th)
end while
if C[available_ind]s == sca_r[available_ind] then
    return s
end if
```

Based on this principle, we introduce the Constrained Optimization-Based Residual Analysis (COBRA) method. This approach efficiently separates correct equations from a error-laden dataset to accomplish the derivation of the private key. The process, outlined in the pseudo-code of Alg.2, operates with inputs such as the coefficient matrix $\mathbf{A}$ derived from the challenge $\mathbf{c}$, results from the side-channel attack denoted as sca_r, solution range $bounds$, error threshold e_{th}, a delta for incrementing error weights each iteration d_{num}, a retention threshold for the remaining equations r_{num}, and a selection parameter for

equations each iteration s_{num}. The algorithm begins by assigning an error weight of zero to all equations and initializing the index for the set of equations.

The core of the algorithm is an iterative process. At each iteration, it randomly selects a subset of equations and uses their coefficients along with the outcomes of the practical attack to compute an approximate solution through constrained optimization. After finding this solution, the algorithm calculates residuals and updates the error weights for each equation. Higher weights suggest a greater probability of errors in the equations, making them candidates for removal in subsequent iterations. This process continues until the number of equations that remain falls below a certain threshold. A solution is considered accurate when the remaining set of equations aligns perfectly with the solution, evidenced by zero residuals, thus indicating successful recovery of the correct solution.

Critical to the algorithm's success are parameters like the error threshold e_{th} and error weights d_{num}, which play pivotal roles in optimizing the balance between the efficiency and stability of the attack. The randomness introduced in each iteration through the selection count of equations s_{num} is essential for handling cases where only part of the private key coefficients can be recovered. In general, when the equation set contains a sufficient number of correct equations, the COBRA algorithm is capable of rapidly and accurately recovering the complete private key.

Typically, solving a system of equations with 256 unknowns necessitates an equivalent number of equations to uniquely determine the solution. However, the distinct distribution of the coefficient matrix combined with constraints on $\mathbf{s}_1$ coefficients uniquely positions us to recover the complete private key using fewer numbers of equations. We found that even if only 200 correct equations are obtained, the complete private key can be recovered in a few seconds using ILP or optimal solving. This means that the information is redundant, and it is still possible to recover the full private key quickly despite some errors in the set of 256 equations provided by a single signature.

4.2 Big-M with Constrained Optimization-Based Residual Analysis

The current challenge is to find a solution, $\mathbf{s}^*$, that maximizes the number of correct equations within a system of $\mathbf{cs}_1$. Marzougui *et al.* [MUTS22] propose addressing this by transforming it into an ILP problem through the Big-M method.

In practice, when the equation system contains a relatively small number of correct equations, the constrained optimization-based residual analysis algorithm might not recover the complete private key. However, many of the recovered error coefficients are usually very close to the correct value, with a deviation of only ± 1. If attackers can determine which coefficients are correct—perhaps through methods like majority voting—they can significantly reduce the key space. Applying the Big-M method could then enable full private key recovery.

If a single run of Alg.2 fails to recover the complete private key, repeat execution of its core computational module may yield alternative solutions due to the random selection of indexes in each iteration to introduce randomness and thus produce different results. For Dilithium2, by statistically analyzing each coefficient's occurrences for ± 2, ± 1, and 0—resulting in five possible outcomes—and setting threshold criteria, the solution space CRF_j (Coefficient Recovery Field) for each coefficient $\mathbf{s}_j$ is defined. When a larger set of original equations is available, typically only 2-3 values stand out within each CRF_j. Notably, with a single signature, which equals merely 256 equations, a more cautious strategy preserves the outcomes of each analysis as potential candidate values.

Fig.2 depicts the optimized Big-M method used in this paper. In comparison to previous studies [MUTS22, BVC+23], we refine constraint (4), previously allowing $\mathbf{s}_j^*$ to range within$\{-\eta, \ldots, \eta\}$, necessitating the assessment of each coefficient against 5 potential outcomes for Dilithium2 and 9 for Dilithium3. Our methodology effectively reduces the range of possible values, thus increasing the algorithm's efficiency.

$$
\begin{aligned}
\text{maximize} \quad & \sum_{i=0}^{|I|-1} \mathbf{x}_i \\
\text{subject to} \quad & \mathbf{x}_i - \mathbf{C}_i\mathbf{s}^* \leq K \cdot (1 - \mathbf{x}_i), && \forall i \in \{0, \ldots, |I| - 1\} \quad (1) \\
& \mathbf{x}_i - \mathbf{C}_i\mathbf{s}^* \geq -K \cdot (1 - \mathbf{x}_i), && \forall i \in \{0, \ldots, |I| - 1\} \quad (2) \\
& \mathbf{x}_i \in \{0, 1\}, && \forall i \in \{0, \ldots, |I| - 1\} \quad (3) \\
& \mathbf{s}_j^* \in CRF_j, && \forall j \in \{0, \ldots, n - 1\} \quad (4)
\end{aligned}
$$

Figure 2: Optimized ILP formulation used for recovering noisy equation system of **cs**.

4.3 Alternative Attack Strategies

Solving Equations using BP. Soft Analytical Side-Channel Attacks (SASCA) is designed to decrease guessing entropy by leveraging side-channel leakages at various points during the execution of an algorithm. It has been successfully applied in attacks on conventional cryptographic systems [VGS14, GGSB20, LWL+22]. A key strategy of this approach involves the use of the BP algorithm, which simplifies global marginalization to local marginalization and employs message passing until convergence, revealing the marginal probability of the targeted value. The BP algorithm is notably effective in post-quantum cryptography attacks, such as those on Kyber [PPM17, PP19, HHP+21].

Bronchain *et al.* [BAE+23] suggested the BP algorithm's application in solving error-containing integer linear equations related to $\mathbf{cs}_1$. The algorithm benefits from the trait that $\mathbf{c}$ comprises only τ nonzero coefficients (either 1 or -1), which simplifies the factor graph and boosts efficiency.

The BP algorithm needs to rely on more equations when the success rate of the practical attack is low. As the dimension increases, the size of the factor graph expands, and the propagation process is more prone to computational overflow problems. We have temporarily failed to complete the solution when the number of equations is excessive. The algorithm also depends on integrating probability distributions of potential values at leakage points into the factor graph, which may poses limitations for attack techniques that don't provide such distributions, like fault and cache attacks. The BP algorithm, being probabilistic in nature, cannot guarantee a solution in every instance. For the system of equations under consideration, determining the conditions under which the algorithm can stabilize and converge to the correct result requires further in-depth study.

Reduction to a LWE Problem. A natural strategy for recovering the entire private key, when partial coefficients are known, involves reducing it to a specialized LWE problem. This situation, where the positions of the recovered coefficients are known, can be considered a type of leaky LWE problem. It is addressed using the leaky LWE estimator proposed by Dachman-Soled *et al.* [DDGR20]. In this case, the known coefficients are integrated into the lattice basis as perfect hints, following which the remaining coefficients are retrieved using lattice reduction techniques such as BKZ. According to the results in [MN23], merely 45% of coefficients are sufficient to break Dilithium2 within 7 days. However, since we cannot determine which coefficients are recovered after solving the erroneous equations $\mathbf{cs}_1 = \mathbf{b}$, this method fails in this context.

When the positions are unknown, the problem of recovering the entire private key with known partial coefficients can be reduced to a ternary LWE problem. Let $\mathbf{s}_1^*$ be the estimator solved by the erroneous equations $\mathbf{cs}_1 = \mathbf{b}$ and substituting it into the public key $\mathbf{t} = \mathbf{As}_1 + \mathbf{s}_2$, we can obtain a new LWE problem $\mathbf{t}' = \mathbf{As}_1' + \mathbf{s}_2$ where $\mathbf{t}' = \mathbf{t} - \mathbf{As}_1^*$,

Here we assume that the public is not compressed or it is reconstructed from a small number of signatures. If it is compressed as done in Dilithium, the new ternary LWE problem is $\mathbf{t}' = \mathbf{As}_1' + \mathbf{e}$ where $\mathbf{e} = \mathbf{s}_2 - \mathbf{t}_0$, $\mathbf{t}_0$ denotes the low order bits of $\mathbf{t}$.

$\mathbf{s}_1' = \mathbf{s}_1 - \mathbf{s}_1^*$. If most equations in $\mathbf{cs}_1 = \mathbf{b}$ are correct, the new problem is likely to be a sparse, ternary LWE problem sharing the same dimension as the original one. According to results in [May21, GM23], the heuristic time/memory complexities is $O(2^{0.345N})$. For Dilithium2, where $N = 1024$, this method appears impractical. We will explore the possibility of combining this method with ILP techniques in the future work.

5 Experiments and Results

5.1 Set Up

Our experiments are conducted on the ChipWhisperer UFO target platform, chosen for its flexibility in supporting various microprocessor modules. We select the STM32F405RGTx microprocessor for its implementation, as per the NIST reference [ACD+22]. To capture leakages, our setup includes a BLP-1.9+ 50M low-pass filter, a PA303 preamplifier, and a WR610Zi oscilloscope, all synchronized to a sampling rate of 100 MSa/s. Although filters like the 1.9+ 5M model, which may offer a higher SNR, are available, we prefer the 1.9+ 50M model to demonstrate the effectiveness of our approach in noisy environments.

We execute the attacks on a desktop computer equipped with an Intel i5-13600KF processor and 32GB of DDR5 RAM. For the machine learning experiments, we utilize 4 TITAN Xp GPUs. This setup ensures sufficient computational power for our analyses. The experimental results we report are the averages from 10 iterations of each experiment, demonstrating the robustness and consistency of our findings across multiple runs.

5.2 LR-SCA for Recovering y

Qiao *et al.* [QLZ+23a] conducted a comparative analysis of power leakages during the operation of generating random numbers y in the Dilithium. Their research pinpointed the "polyz_unpack(a, buf)" function as exhibiting the most significant leakage related to y, thereby designating this function as the focus of our attack. As illustrated in Fig.3, this function is tasked with extracting an 18-bit value from a predefined random array, denoted as r, and converting it into y through the operation GAMMA1-r, where GAMMA1 $= 2^{17}$. This conversion process takes place over 64 cycles, producing four instances of y per cycle.

The analysis indicates a strong correlation between r and y, demonstrating that knowledge of either variable is sufficient for completing the analysis. Fig.4 presents the correlation analysis of the lower 8 bits of r and y using 1,000 traces under the HW leakage model. The results reveal that the leakage from r is more pronounced, possibly due to r undergoing more operations. Consequently, targeting the operations associated with r is likely to yield significantly better results than directly attacking y.

In the Dilithium2 scheme, we utilize 50,000 traces and apply a low-pass filter using Fast Fourier Transform (FFT) to enhance the signal quality. Subsequently, we identify continuous Points of Interest (POIs) using Pearson Correlation Coefficients (PCCs). It is important to note that we are profiling r separately for each coefficient position to prevent potential misalignments from affecting the trace alignment.

In our experiments, we methodically examine the impact of varying the number of PoIs on our attack's effectiveness, focusing on two scenarios based on the distribution of $\mathbf{cs}_1$: either $||\mathbf{cs_1}||_\infty < 2^4$ or $||\mathbf{cs_1}||_\infty < 2^5$. The results, detailed in Tab.4for 256 coefficients profiled and attacked, illustrate the relationship among the number of POIs, the profiling time, and the attack's success rate (SR). Notably, with 20 POIs, the profiling time is 134.2 seconds, achieving a 14.2% under the 4-bit assumption within 7.2 seconds, and an 8.8% SR under the 5-bit assumption in 15.89 seconds. When the POIs increase to 300, the profiling time extends to 952.3 seconds, which significantly boosts the SR to 39.6% for the 4-bit assumption in 240.6 seconds, and to 33.4% ffor the 5-bit assumption in 461.3 seconds.

```
void polyz_unpack(poly *r, const uint8_t *a) {
    for(int i = 0; i < N/4; ++i) {
        r->coeffs[4*i+0]  = a[9*i+0];
        r->coeffs[4*i+0] = (uint32_t)a[9*i+1] << 8;
        r->coeffs[4*i+0] |= (uint32_t)a[9*i+2] << 16;
        r->coeffs[4*i+0] &= 0x3FFFF;

        r->coeffs[4*i+1] = a[9*i+2] >> 2;
        r->coeffs[4*i+1] |= (uint32_t)a[9*i+3] << 6;
        r->coeffs[4*i+1] |= (uint32_t)a[9*i+4] << 14;
        r->coeffs[4*i+1] &= 0x3FFFF;

        r->coeffs[4*i+2] = a[9*i+4] >> 4;
        r->coeffs[4*i+2] |= (uint32_t)a[9*i+5] << 4;
        r->coeffs[4*i+2] |= (uint32_t)a[9*i+6] << 12;
        r->coeffs[4*i+2] &= 0x3FFFF;

        r->coeffs[4*i+3] = a[9*i+6] >> 6;
        r->coeffs[4*i+3] |= (uint32_t)a[9*i+7] << 2;
        r->coeffs[4*i+3] |= (uint32_t)a[9*i+8] << 10;
        r->coeffs[4*i+3] &= 0x3FFFF;

        r->coeffs[4*i+0] = GAMMA1 - r->coeffs[4*i+0];
        r->coeffs[4*i+1] = GAMMA1 - r->coeffs[4*i+1];
        r->coeffs[4*i+2] = GAMMA1 - r->coeffs[4*i+2];
        r->coeffs[4*i+3] = GAMMA1 - r->coeffs[4*i+3];}}
```

Figure 3: Polyz_unpack(a, buf) reference implementation.

Table 4: Results for Recovering 256 Coefficients of y

#PoI	Profiling Time (s)	Time[1] (s)	SR[1] (%)	Time[2] (s)	SR[2] (%)
20	134.2	7.2	14.2	15.89	8.8
50	170.0	21.4	16.4	36.7	10.8
100	321.4	56.4	24.1	121.6	17.3
300	952.3	240.6	39.6	461.3	33.4

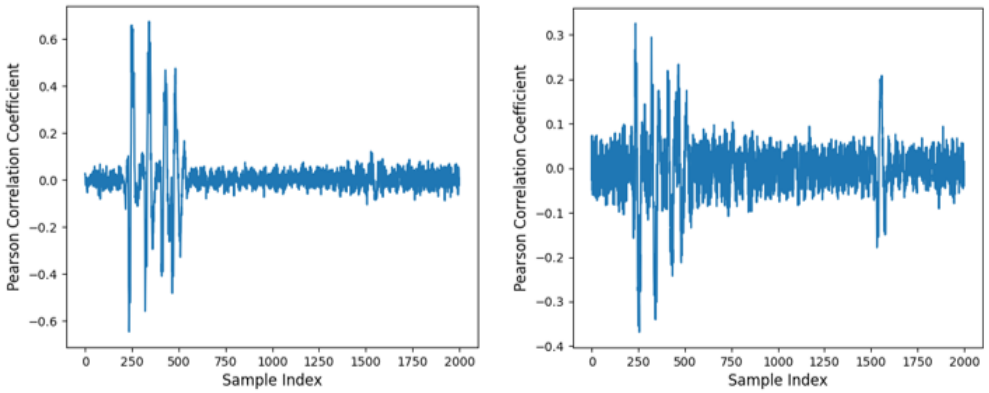
[1] assumption for $||\mathbf{cs_1}||_\infty < 2^4$.
[2] assumption for $||\mathbf{cs_1}||_\infty < 2^5$.

The observed variations in the success rates of our attacks are significantly influenced by the value range and distribution characteristics of $\mathbf{cs}_1$. The distribution of $\mathbf{cs}_1$ closely approximates a Gaussian curve, with approximately 95% of the data falling within a 4-bit range. The attack already covers the vast majority of possibilities, thereby reducing the enumeration space and consequently enhancing the attack's success rate. Conversely, extending the assumption to a 5-bit range increases the value space, capturing only an additional 5% of possible values but at the cost of doubling the value space. Although this extension might seem to allow for the recovery of a broader range of values, it actually diminishes the attack's effectiveness.

Our method demonstrates the capability to fully recover the value of y with success rates nearing 40% using a single trace, representing a significant advancement in efficiency compared to previous approaches. This improvement is primarily attributed to the reduction in the number of traces needed to build templates and a smaller enumeration space during template matching. Crucially, our attack does not require the selection of a special y values, thereby further enhancing its efficiency.

5.3 DL-SCA for Recovering $\mathbf{cs}_1$

Chen *et al.* [CKA+21] identified significant leakage in the Montgomery reduction operation, a critical component of the Dilithium algorithm's reference implementation submitted to



(a) PCC of r (b) PCC of y

Figure 4: PCC of r and y with 1,000 traces

NIST. This operation is essential to the entire INTT process, facilitating the conversion of results from the NTT domain back to the normal domain for subsequent calculations. The specific implementation of the Montgomery reduction is illustrated in Fig.5. Notably, the reduction process, especially the shifting and storage operations in its final stages, is recognized as a likely primary source of significant $\mathbf{cs}_1$ leakage. We targeted these operations, achieving a maximum SNR of 16 in this experiment.

```
int32_t montgomery_reduce(int64_t a) {
    int32_t t;

    t = (int32_t)a*QINV;
    t = (a - (int64_t)t*Q) >> 32;

    return t;}
```

Figure 5: Montgomery_reduce reference implementation.

Dilithium offers both deterministic and randomized schemes. The deterministic scheme allows for the recovery of all discarded challenge $\mathbf{c}$ values when the private key is known, a feat not possible with the randomized scheme. In our analysis, we specifically focus on leakages from the final round of legitimate signatures.For Dilithium2, we capture the complete round of $\mathbf{cs}_1$ leakage. While theoretically, collecting a broader dataset by including $\mathbf{cs}_2$ leakages could provide more data for training, our current discussion remains exclusively focused on $\mathbf{cs}_1$, excluding $\mathbf{cs}_2$ from our scope.

For Dilithium2, each signature provides 1,024 training samples. We statically align and segment these samples into blocks of 400 for training the CNN model, as detailed in Sec.3.2. Our training strategy addresses the range of $\mathbf{cs}_1$, focusing on data within $||\mathbf{cs_1}||_\infty < 2^4$, $||\mathbf{cs_1}||_\infty < 2^5$ and $||\mathbf{cs_1}||_\infty < \beta$. Using about 200,000 samples provided by 200 signatures, it takes about 40 minutes to complete the training.

Fig.6 illustrates the guess entropy and success rates for side-channel attacks under various strategies. Using consistent model hyperparameters, expanding the training set enables the creation of more accurate classifiers. Generally, assuming $||\mathbf{cs_1}||_\infty < 2^5$ slightly surpassese $||\mathbf{cs_1}||_\infty < 2^4$ assumption, with success rates improving from 70% to 74%. For guess entropy, the assumption of $||\mathbf{cs_1}||_\infty < 2^5$ approaches 1. The reason for similar success rates but significant differences in guess entropy is that under the $||\mathbf{cs_1}||_\infty < 2^4$ assumption, some actual values fall outside our hypothesized space and are ranked last in our analysis. These findings highlight the potential of CNN-based techniques, suggesting

that further optimization of network hyperparameters might enhance success rates.

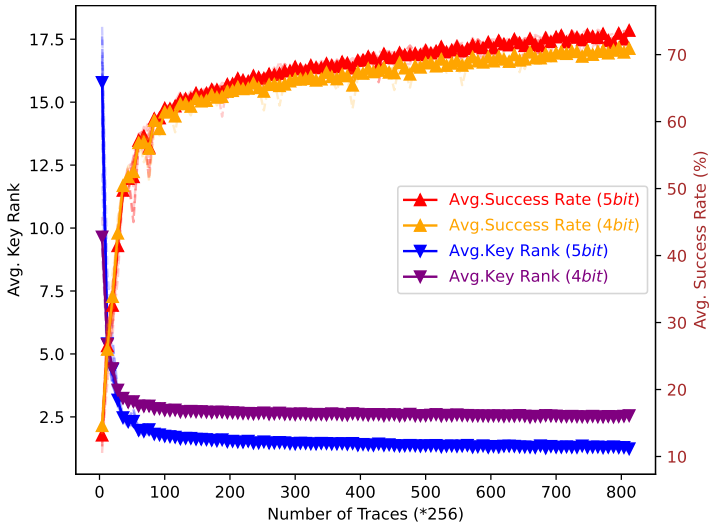


Figure 6: Guess entropy and success rates of $\mathbf{cs}_1$

Furthermore, we compared our method with traditional TAs for a comprehensive analysis. In the TAs, POIs were identified based on the top 20, 50, and 100 PCCs, as well as attempts to use all available points. The success rates, as detailed in Tab.5, demonstrate a trend where an increased number of POIs generally enhances the attack's effectiveness. When employing all feature points, TAs achieved success rates of 54.6% and 57.7% under the assumptions of $||\mathbf{cs_1}||_\infty < 2^4$ and $||\mathbf{cs_1}||_\infty < 2^5$, respectively. Although TAs also yield better results with more PoIs they remain inferior to CNN-based attacks, likely due to misalignments in the static alignment process. Consequently, CNN-based methods have become the preferred approach for analyzing and exploiting $\mathbf{cs}_1$ leakage during the Montgomery reduction process.

Table 5: Success Rates of CNN and TA

Approach	**SR of** $||\mathbf{cs_1}||_\infty < 2^4$	**SR of** $||\mathbf{cs_1}||_\infty < 2^5$	**SR of** $||\mathbf{cs_1}||_\infty < \beta$
CNN	70.5	74.3	70.6
TA (#PoI=20)	25.9	23.1	22.9
TA (#PoI=50)	28.7	22.6	22.6
TA (#PoI=100)	40.5	40.4	31.2
TA (#PoI=400)	54.6	57.7	57.5

It should also be noted that for the private key $\mathbf{s}_2$, the computation $\mathbf{cs}_2$, which is involved, undergoes the same operation as $\mathbf{cs}_1$. Therefore, it is entirely feasible to apply the same method to conduct an actual attack on $\mathbf{cs}_2$ if required.

Moreover, we employ the method described in Sec.3.3 to integrate the results of $\mathbf{cs}_1$ with y. Tab.6 presents the success rates achieved, showcasing the impact of varying the PoIs for y. Assuming $||\mathbf{cs_1}||_\infty < 2^4$, we achieve an optimal success rate of 86%. This rate further increases to an average of 92.8% when the model assumes $||\mathbf{cs_1}||_\infty < 2^5$. These results compellingly demonstrate the significantly improved effectiveness of the combined attack strategy.

Table 6: Success Rates for Combined y and cs_1 Results

Range of $\mathbf{cs}_1$	**$\mathbf{cs}_1$ SR (%)**	**#PoI of y**	**y SR (%)**	**Merged SR (%)**
$\|\|\mathbf{cs_1}\|\|_\infty < 2^4$	70.5	20	14.2	80.9
		50	16.4	82.3
		100	24.1	83.1
		300	39.6	86.7
$\|\|\mathbf{cs_1}\|\|_\infty < 2^5$	74.3	20	8.8	84.9
		50	10.8	86.5
		100	17.3	88.7
		300	33.4	92.8

5.4 Constrained Optimization-Based Residual Analysis Result

In our practical experiments, we observed that when a set of 256 equations contains more than 8 errors, it is not possible to recover the private key within one hour using the ILP on our computing device, with the time cost growing exponentially as the number of errors increases. We evaluated the number of equations required to fully recover the private key using residual analysis based on constrained optimization across various proportions of correct equations, thus providing guidance for actual attacks.

Table 7: Performance of COBRA

Correct Eq. Ratio	**Eq. Count ($\times$256)**	$\mathbf{d_{num}}$	$\mathbf{e_{th}}$	**SR (%)**	**Time (s)**
10%	65	50	0	100	189.5
30%	9	50	10	100	53.2
50%	4	50	5	100	10.0
70%	3	40	0	100	1.63
90%	2	30	0	100	0.8
95%	1	2	2	83	4.1

Tab.7 details the necessary equations and specific parameter adjustments required for successful private key recovery, given varying percentages of correct equations. It demonstrates that private key recovery is feasible across different success rates, provided a certain threshold number of equations is met. Generally, the number of required equations decreases as the success rate increases. This parameterization is heuristic, designed for flexibility to allow real-time adjustments that enhance practical implementation efficiency. As the proportion of incorrect equations increases, d_{num} can be adjusted upward to accelerate the elimination of these inaccurate equations. Although 230 correct equations, denoted as r_{num}, are sufficient for private key recovery, adjusting the parameters to fit the actual situation can further increase the speed of the attack.

In practice, if the speed of the attack is not a priority, the number of required equations can be further reduced by adjusting the parameters. For instance, in scenarios where only 10% of the equations are correct, we successfully executed the attack by setting e_{th} to 5, using fewer than 60×256 equations (corresponding to 60 signatures). If attackers have access to a large set of equations, it is advisable to increase the number of equations to expedite the resolution process. However, setting d_{num} too high should be avoided, as it may inadvertently eliminate a significant number of correct equations early in the process.

Our approach has shown the ability to quickly find solutions even when the set of 256 equations contains inaccuracies. In our experiments, we randomly generated 100 sets of equations with a 95% accuracy rate. After applying our algorithm 10 times on each set, approximately 83% of these equation sets successfully led to the recovery of the complete private key $\mathbf{s}_1$. This approach has been more efficient than conventional ILP solutions.

Referring to the simulation experiments under the HW model as described by [BVC+23], we explore the number of signatures required for COBRA to recover the private key under various SNR ranges and $\mathbf{cs_1}$ assumptions. For this simulation, we selected 20 PoIs, profiled

Table 8: Experimental results of COBRA simulation with different SNRs

SNR	Range of $\mathbf{cs}_1$	$\mathbf{cs}_1$ SR (%)	Eq. Count ($\times$256)
0.01	$\|\|\mathbf{cs_1}\|\|_\infty < 2^4$	5.1	220
	$\|\|\mathbf{cs_1}\|\|_\infty < 2^5$	3.1	500
	$\|\|\mathbf{cs_1}\|\|_\infty < \beta$	2.3	650
0.1	$\|\|\mathbf{cs_1}\|\|_\infty < 2^4$	7.8	95
	$\|\|\mathbf{cs_1}\|\|_\infty < 2^5$	5.5	200
	$\|\|\mathbf{cs_1}\|\|_\infty < \beta$	3.9	440
1	$\|\|\mathbf{cs_1}\|\|_\infty < 2^4$	13.7	36
	$\|\|\mathbf{cs_1}\|\|_\infty < 2^5$	10.9	52
	$\|\|\mathbf{cs_1}\|\|_\infty < \beta$	9.0	77
10	$\|\|\mathbf{cs_1}\|\|_\infty < 2^4$	33.6	8
	$\|\|\mathbf{cs_1}\|\|_\infty < 2^5$	30.9	9
	$\|\|\mathbf{cs_1}\|\|_\infty < \beta$	29.9	9
100	$\|\|\mathbf{cs_1}\|\|_\infty < 2^4$	64.5	3
	$\|\|\mathbf{cs_1}\|\|_\infty < 2^5$	67.2	3
	$\|\|\mathbf{cs_1}\|\|_\infty < \beta$	66.4	3

using 50,000 traces, and attacked y and $\mathbf{cs_1}$ separately to obtain HW values. These values were then merged according to the method outlined by [BVC+23]. The combined part of the result produces multiple candidates. We multiplied the probability value of each candidate according to the data distribution of $\mathbf{cs_1}$and selected the maximum value as the final $\mathbf{cs_1}$ result. This experiment was repeated 50 times, with e_{th} set to zero. Tab.8 presents the success rate of $\mathbf{cs_1}$ and the number of signatures required to recover $\mathbf{s}_1$. With an SNR of 100, all three $\mathbf{cs_1}$ assumptions could complete the attack with just 3 signatures. The assumption $||\mathbf{cs_1}||_\infty < 2^4$ provides the best results at an SNR of 0.01, with only 5% correctness, but the attack can be conducted with 220 signatures.

In real-world analysis, when the success rate is low, this method efficiently eliminates a large number of erroneous equations in the initial phase. Although it may not yield the correct result immediately, it serves as an effective precursor operation to BP or ILP, significantly improving the efficiency of the attack.

5.5 Practical SCAs of Dilithium

Next, we present the practical attack results for Dilithium2. Considering real-world scenarios where attackers may only be able to obtain leakage for y and $\mathbf{cs}_1$ separately, this section will detail the complete attack outcomes in three scenarios: attacking y alone, $\mathbf{cs}_1$ alone, and simultaneously obtaining leakage information for both y and $\mathbf{cs}_1$.

Table 9: Signatures Required for Private Key Recovery

Range of $\mathbf{cs}_1$	Side-Channel Strategy	#Signatures
$\|\|\mathbf{cs_1}\|\|_\infty < 2^4$	Attack on y (#POI=300)	8(6)
	Attack on $\mathbf{cs_1}$	3(3)
	Hybrid	2(2)
$\|\|\mathbf{cs_1}\|\|_\infty < 2^5$	Attack on y (#POI=300)	9(7)
	Attack on $\mathbf{cs_1}$	3(2)
	Hybrid	2(1)

() corresponds to the best case that occurs in the attack.

Table 9 documents the number of signatures needed for private key recovery under different scenarios, showcasing both the highest and lowest numbers of signatures required in successful attempts across a series of 10 experiments. The efficiency of the attack demonstrates minimal fluctuation across different assumptions about the bit range of

$\mathbf{cs}_1$. Targeting the leakage of y requires approximately 10 signatures to recover $\mathbf{s}_1$. In contrast, focusing solely on the leakage from $\mathbf{cs}_1$ allows for completing the attack with just 3 signatures, thanks to the high success rates achieved by the CNN model. In cases assuming $||\mathbf{cs_1}||_\infty < 2^5$, only 2 signatures are needed. Exploiting leakage from both y and $\mathbf{cs}_1$ requires at most 2 signatures to recover $\mathbf{s}_1$. Notably, our empirical analysis revealed that about 20% of cases could succeed with leakage from a single signature, typically requiring the recovery of 240 out of 256 coefficients through the side-channel attack.

The results above provide strong evidence of the effectiveness of our scheme, demonstrating that the number of signatures required to complete the attack can be kept under 10 even when targeting only a random number y with a low success rate. This represents a significant improvement over previous work by Marzougui *et al.* [MUTS22] and Berzati *et al.* [BVC+23]. Furthermore, our practical attack confirms that recovering Dilithium's private key with just one signature is indeed feasible.

For a single signature, we utilized the optimized ILP method described in in Sec.4.2, to perform the attack. Initially, we conducted 1,000 repetitions under a constrained optimization algorithm based on residual analysis to calculate possible values for $\mathbf{cs}_1$, typically completing this process within 20 minutes. Following this preparatory phase, we applied the optimized Big-M method for the final solution. Observations show that when the side-channel attack successfully recovers more than 236 equations, the ILP phase generally concludes within 30 minutes. If the number of correctly recovered equations is below this threshold, the solution time often extends to over an hour. This result significantly exceeds the capabilities of using ILP alone, which tolerates at most 8 errors within a similar timeframe, thereby greatly improving solution efficiency. In this experimental framework, about 60% of the attacks recover the private key within one hour for single-signature scenarios.

6 Conclusion and future work

In this study, we conducted a comprehensive side-channel analysis of Dilithium2, focusing on the polynomial addition operation $z = y + \mathbf{cs}_1$. Utilizing LR-based profiled attacks, we achieved a 40% success rate in recovering the complete value of y, and with the aid of a CNN model, we succeeded in recovering the value of $\mathbf{cs}_1$ with a 75% success rate. By integrating these findings, we enhanced the success rate for recovering $\mathbf{cs}_1$ through side-channel analysis to 92%. Furthermore, we introduced a constrained optimization-based residual analysis, enabling the swift recovery of the private key $\mathbf{s}_1$ from extensive sets of $\mathbf{cs}_1$ equations, even those containing errors. The results from actual attacks on Dilithium2 indicate that our approach can efficiently recover the private key $\mathbf{cs}_1$ with minimal leakage from generated signatures—in the optimal scenario, requiring only a single signature, with comparatively low time overhead.

Given that $\mathbf{cs}_2$ also undergoes the Montgomery reduction operation, our method theoretically extends to the recovery of $\mathbf{s}_2$, albeit necessitating approximately 2-3 signatures due to the lack of y to bolster the attack's efficacy.

Despite the substantial success in recovering most of $\mathbf{cs}_1$, a challenge persists: even after recovering 230 coefficients and applying our constrained optimization-based residual analysis to reduce constraints, the computational burden of solving through ILP remains substantial. We suggest that integrating our side-channel findings with the BP algorithm could enable a more consistent realization of the 1-signature attack. In future efforts, we aim to explore more efficient mathematical methods to achieve Dilithium attacks under single signatures with improved stability and efficiency. Additionally, we plan to investigate the effectiveness of our approach against protected implementations of Dilithium. Given that constrained optimization-based residual analysis is still applicable with a 5% success rate of side-channel attacks, we believe it will be highly beneficial for protected implementations.

References

[ACD+22] Gorjan Alagic, David Cooper, Quynh Dang, Thinh Dang, John M. Kelsey, Jacob Lichtinger, Yi-Kai Liu, Carl A. Miller, Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson, Daniel Smith-Tone, and Daniel Apon. Status report on the third round of the nist post-quantum cryptography standardization process, 2022-07-05 04:07:00 2022.

[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. *IACR Cryptol. ePrint Arch.*, page 1545, 2023.

[BCL99] Mary Ann Branch, Thomas F. Coleman, and Yuying Li. A subspace, interior, and conjugate gradient method for large-scale bound-constrained minimization problems. *SIAM J. Sci. Comput.*, 21(1):1–23, 1999.

[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: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings*, volume 3156 of *Lecture Notes in Computer Science*, pages 16–29. Springer, 2004.

[BJL+14] Aurélie Bauer, Éliane Jaulmes, Victor Lomné, Emmanuel Prouff, and Thomas Roche. Side-channel attack against RSA key generation algorithms. In Lejla Batina and Matthew Robshaw, editors, *Cryptographic Hardware and Embedded Systems - CHES 2014 - 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings*, volume 8731 of *Lecture Notes in Computer Science*, pages 223–241. Springer, 2014.

[BVC+23] Alexandre Berzati, Andersson Calle Viera, Maya Chartouny, Steven Madec, Damien Vergnaud, and David Vigilant. Exploiting intermediate value leakage in dilithium: A template-based approach. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2023(4):188–210, 2023.

[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 *39th IEEE International Conference on Computer Design, ICCD 2021, Storrs, CT, USA, October 24-27, 2021*, pages 583–590. IEEE, 2021.

[CRR02] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*, volume 2523 of *Lecture Notes in Computer Science*, pages 13–28. Springer, 2002.

[DDGR20] Dana Dachman-Soled, Léo Ducas, Huijing Gong, and Mélissa Rossi. LWE with side information: Attacks and concrete security estimation. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part II*, volume 12171 of *Lecture Notes in Computer Science*, pages 329–358. Springer, 2020.

[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 Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(1):238–268, 2018.

[FDK20] Apostolos P. Fournaris, Charis Dimopoulos, and Odysseas G. Koufopavlou. Profiling dilithium digital signature traces for correlation differential side channel attacks. In Alex Orailoglu, Matthias Jung, and Marc Reichenbach, editors, *Embedded Computer Systems: Architectures, Modeling, and Simulation - 20th International Conference, SAMOS 2020, Samos, Greece, July 5-9, 2020, Proceedings*, volume 12471 of *Lecture Notes in Computer Science*, pages 281–294. Springer, 2020.

[GGSB20] Qian Guo, Vincent Grosso, François-Xavier Standaert, and Olivier Bronchain. Modeling soft analytical side-channel attacks from a coding theory viewpoint. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2020(4):209–238, 2020.

[GM23] Timo Glaser and Alexander May. How to enumerate LWE keys as narrow as in kyber/dilithium. In Jing Deng, Vladimir Kolesnikov, and Alexander A. Schwarzmann, editors, *Cryptology and Network Security - 22nd International Conference, CANS 2023, Augusta, GA, USA, October 31 - November 2, 2023, Proceedings*, volume 14342 of *Lecture Notes in Computer Science*, pages 75–100. Springer, 2023.

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

[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.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 1999.

[Koc96] Paul C. Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In Neal Koblitz, editor, *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer, 1996.

[LWL+22] Sinian Luo, Weibin Wu, Yanbin Li, Ruyun Zhang, and Zhe Liu. An efficient soft analytical side-channel attack on ascon. In Lei Wang, Michael Segal, Jenhui Chen, and Tie Qiu, editors, *Wireless Algorithms, Systems, and Applications - 17th International Conference, WASA 2022, Dalian, China, November 24-26, 2022, Proceedings, Part I*, volume 13471 of *Lecture Notes in Computer Science*, pages 389–400. Springer, 2022.

[LZS+21] Yuejun Liu, Yongbin Zhou, Shuo Sun, Tianyu Wang, Rui Zhang, and Jingdian Ming. On the security of lattice-based fiat-shamir signatures in the presence of randomness leakage. *IEEE Trans. Inf. Forensics Secur.*, 16:1868–1879, 2021.

[May21] Alexander May. How to meet ternary LWE keys. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part II*, volume 12826 of *Lecture Notes in Computer Science*, pages 701–731. Springer, 2021.

[MN23] Alexander May and Julian Nowakowski. Too many hints - when LLL breaks LWE. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology - ASIACRYPT 2023 - 29th International Conference on the Theory and Application of Cryptology and Information Security, Guangzhou, China, December 4-8, 2023, Proceedings, Part IV*, volume 14441 of *Lecture Notes in Computer Science*, pages 106–137. Springer, 2023.

[MPP16] Houssem Maghrebi, Thibault Portigliatti, and Emmanuel Prouff. Breaking cryptographic implementations using deep learning techniques. In Claude Carlet, M. Anwar Hasan, and Vishal Saraswat, editors, *Security, Privacy, and Applied Cryptography Engineering - 6th International Conference, SPACE 2016, Hyderabad, India, December 14-18, 2016, Proceedings*, volume 10076 of *Lecture Notes in Computer Science*, pages 3–26. Springer, 2016.

[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. *IACR Cryptol. ePrint Arch.*, page 106, 2022.

[Ols04] Loren D. Olson. Side-channel attacks in ECC: A general technique for varying the parametrization of the elliptic curve. In Marc Joye and Jean-Jacques Quisquater, editors, *Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings*, volume 3156 of *Lecture Notes in Computer Science*, pages 220–229. Springer, 2004.

[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 - 6th International Conference on Cryptology and Information Security in Latin America, Santiago de Chile, Chile, October 2-4, 2019, Proceedings*, volume 11774 of *Lecture Notes in Computer Science*, pages 130–149. Springer, 2019.

[PPM17] Robert Primas, Peter Pessl, and Stefan Mangard. Single-trace side-channel attacks on masked lattice-based encryption. In Wieland Fischer and Naofumi Homma, editors, *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, volume 10529 of *Lecture Notes in Computer Science*, pages 513–533. Springer, 2017.

[QLZ+23a] Zehua Qiao, Yuejun Liu, Yongbin Zhou, Jingdian Ming, Chengbin Jin, and Huizhong Li. Practical public template attack attacks on crystals-dilithium with randomness leakages. *IEEE Trans. Inf. Forensics Secur.*, 18:1–14, 2023.

[QLZ+23b] Zehua Qiao, Yuejun Liu, Yongbin Zhou, Mingyao Shao, and Shuo Sun. When NTT meets SIS: efficient side-channel attacks on dilithium and kyber. *IACR Cryptol. ePrint Arch.*, page 1866, 2023.

[QS01] Jean-Jacques Quisquater and David Samyde. Electromagnetic analysis (EMA): measures and counter-measures for smart cards. In Isabelle Attali and

Thomas P. Jensen, editors, *Smart Card Programming and Security, International Conference on Research in Smart Cards, E-smart 2001, Cannes, France, September 19-21, 2001, Proceedings*, volume 2140 of *Lecture Notes in Computer Science*, pages 200–210. Springer, 2001.

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 124–134. IEEE Computer Society, 1994.

[SLP05] Werner Schindler, Kerstin Lemke, and Christof Paar. A stochastic model for differential side channel cryptanalysis. In Josyula R. Rao and Berk Sunar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings*, volume 3659 of *Lecture Notes in Computer Science*, pages 30–46. Springer, 2005.

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

[VGS14] Nicolas Veyrat-Charvillon, Benoît Gérard, and François-Xavier Standaert. Soft analytical side-channel attacks. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, volume 8873 of *Lecture Notes in Computer Science*, pages 282–296. Springer, 2014.

[WAGP20] Lennert Wouters, Victor Arribas, Benedikt Gierlichs, and Bart Preneel. Revisiting a methodology for efficient CNN architectures in profiling attacks. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2020(3):147–168, 2020.

[WNGD23] Ruize Wang, Kalle Ngo, Joel Gärtner, and Elena Dubrova. Single-trace side-channel attacks on crystals-dilithium: Myth or reality? *IACR Cryptol. ePrint Arch.*, page 1931, 2023.

[ZBHV20] Gabriel Zaid, Lilian Bossuet, Amaury Habrard, and Alexandre Venelli. Methodology for efficient CNN architectures in profiling attacks. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2020(1):1–36, 2020.