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

Zehua Qiao[1,2], 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

Abstract.
As the National Institute of Standards and Technology (NIST) concludes its post-quantum cryptography (PQC) competition, the winning algorithm, Dilithium, enters the deployment phase in 2024. This phase underscores the importance of conducting thorough practical security evaluations. Our study offers an in-depth side-channel analysis of Dilithium, showcasing the ability to recover the complete private key, $\mathbf{s}_1$, within ten minutes using just two signatures and achieving a 60% success rate with a single signature. We focus on analyzing the polynomial addition in Dilithium, $z = y + \mathbf{cs}_1$, by breaking down the attack into two main phases: the recovery of y and $\mathbf{cs}_1$ through side-channel attacks, followed by the resolution of a system of error-prone equations related to $\mathbf{cs}_1$. Employing Linear Regression-based profiled attacks enables the successful recovery of the full y value with a 40% success rate without the necessity for initial filtering. The extraction of $\mathbf{cs}_1$ is further improved using a CNN model, which boasts an average success rate of 75%. A significant innovation of our research is the development of a constrained optimization-based residual analysis technique. This method efficiently recovers $\mathbf{s}_1$ from a large set of error-containing equations concerning $\mathbf{cs}_1$, proving effective even when only 10% of the equations are accurate. We conduct a practical attack on the Dilithium2 implementation on an STM32F4 platform, demonstrating that typically two signatures are sufficient for complete private key recovery, with a single signature sufficing in optimal conditions. Using a general-purpose PC, the full private key can be reconstructed in ten minutes.

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

1 Introduction

Rapid advancements in quantum computing present a significant threat to cryptographic algorithms that rely on the computational difficulty of problems such as integer factorization and discrete logarithms. Should a general-purpose quantum computer be successfully developed, it is expected that the quantum algorithm proposed by Shor [Sho94] in 1994 would render these cryptographic algorithms vulnerable to being broken in polynomial time. In response, the National Institute of Standards and Technology (NIST) has initiated the PQC competition, which has led to the identification of CRYSTALS-Dilithium (abbr.Dilithium) as a digital signature candidate.

Dilithium [DKL+18] is a digital signature scheme based on the hardness of lattice problems, utilizing the Fiat-Shamir paradigm within the polynomial ring $\mathbb{Z}_{q[x]}/(x^n+1)$. Its

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

recognition among experts is due to its comprehensive performance that marries operational efficiency with theoretical safety. This balance positions Dilithium as a prominent candidate in the field of post-quantum cryptography, selected for its capability to secure digital communications against the quantum computing threat.

Despite the theoretical resilience of PQC algorithms against both quantum and classical computational attacks, their real-world implementations remain vulnerable to side-channel analysis. This form of attack exploits unintentional 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 matured significantly within the field of cryptanalysis, marking substantial achievements. It has successfully facilitated the practical analysis of cryptographic algorithms, including those targeting DES [KJJ99], AES [BCO04], RSA [BJL+14], and ECC [Ols04], among others.

To date, numerous studies have been conducted to evaluate the security of Dilithium implementations, yielding significant findings. In the realm of side-channel analysis, particularly the non-template class exemplified by Correlation Power Analysis (CPA), such investigations have predominantly focused on the polynomial multiplication operation $\mathbf{cs}_1$ for conducting attacks. Chen *et al.* [BCO04], building upon the methodology proposed by Fournaris *et al.* [FDK20] for the Montgomery reduction operation, have demonstrated that the CPA technique, aimed at the exhaustive recovery of the private key from 157 traces, can achieve this in 6,357 seconds for one Number Theoretic Transform (NTT) domain coefficient. Furthermore, the integration of a partitioning method has been shown to accelerate the attack by a factor of 7.77. Qiao *et al.* have developed a CPA-based attack, enhanced with the LLL algorithm, to accomplish full Dilithium private key recovery in less than one minute. Additionally, Liu *et al.* [QLZ+23b] have introduced an innovative random leakage attack strategy, leveraging public template attacks to extract lower-bit polynomial coefficient information. This approach significantly streamlines the private key recovery process, reducing it to a solvable integer LWE problem within polynomial time.

The exploration of profiling side-channel attacks, especially those integrating machine learning methodologies, has provided significant advancements. Han *et al.* [HLK+21] initiated the recovery of all Dilithium private keys by focusing on the NTT's initial butterfly operations during the signature generation phase, employing a machine learning-based template attack. Following this, Marzougui *et al.* [MUTS22] devised an attack targeting the sensitive random number y, correlating sensitive parameters with specific values (e.g., $y = 0$) and constructing a system of linear equations about $\mathbf{cs}_1$ to resolve the Dilithium private key $\mathbf{s}_1$. Berzati *et al.* [BVC+23] honed in on a narrower spectrum of sensitive intermediate values, particularly $\boldsymbol{w}_0$, using a filtering algorithm specifically designed for Dilithium's parameter traits. Wang *et al.* [WNGD23] launched an attack on the secret key unpacking phase of the signing algorithm, leveraging deep learning-assisted profiled power analysis. This approach harbors a slim chance of completing the private key recovery with a single trace, boasting a success rate nearing 100% after 74 signatures. A common challenge among several of these works is the task of solving a system of equations concerning the error-prone $\mathbf{cs}_1$, predominantly utilizing Integer Linear Programming (ILP) for this purpose. Bronchain *et al.* [BAE+23] applied the Belief Propagation (BP) algorithm to solve for polynomial multiplication $\mathbf{cs}_1$ and conducted simulation experiments to attack y. Their optimal finding indicates that recovering the private key $\mathbf{s}_1$ can be achieved with four signatures using the Hamming model at a signal-to-noise ratio (SNR) of 100.

In profiling attacks, particularly those aimed at values such as y and $\boldsymbol{w}_0$, the fundamental strategy is to choose specific numbers to enhance the success rate of side-channel attacks. This approach, however, requires a substantial number of power traces for both the construction and matching of templates due to the large candidate space of the above targets. Moreover, given that these target values are regenerated randomly with each signature, the attack typically hinges on a single trace, which frequently fails to secure a high success rate. When faced with numerous errors, the application of lattice basis reduction algorithms presents its own set of challenges, and alternative approaches, such as ILP, demand considerable time. Bronchain *et al.* [BAE$^+$23] suggested the use of BP algorithm to resolve the system of error-prone equations in Dilithium related to $\mathbf{cs}_1$, though their investigations were confined to simulation experiments. Furthermore, the low accuracy of side-channel information necessitates an increased number of equations, leading to an excessively large graph for the BP algorithm and significantly elevating the risk of computational overflow. Due to the need to determine the probability distribution of all candidate values for the target value, the feasibility of the BP algorithm in attacks such as fault injection, where a probability distribution is not available, remains to be determined.

This paper introduces a novel method for swiftly recovering the Dilithium private key with a reduced number of signatures. We propose an approach for the side-channel attack aspect that enables the complete recovery of the random number y with a high success rate, eliminating the need for any filtering. Additionally, we explore an attack on $\mathbf{cs}_1$ targeting a significantly narrowed value space. Moreover, we unveil a constrained optimization-based residual analysis technique tailored for efficiently solving error-prone linear equations associated with $\mathbf{cs}_1$. These methodologies were deployed in a practical attack scenario against the open-source implementation of Dilithium2, conducted on an STM32F4 platform, which substantiates the feasibility and efficacy of our proposed techniques. The specific contributions of our research are as follows:

- Building upon Qiao *et al.*'s [QLZ$^+$23a] discovery that higher bit information of y can be deduced from the signature z, we introduce a Linear Regression-based side-channel attack. This method effectively mitigates the influence of higher bits, thereby enhancing the attack's success rate. In practical experiments, this approach achieves a 40% success rate in attacking y.

- We exploit the characteristic that $\mathbf{cs}_1$ operates within a limited value range, with each coefficient being independently calculated. For Dilithium2, a single signature unveils a minimum of 1024 data leaks, which are conducive to machine learning pre-training. Employing a Convolutional Neural Networks (CNNs) model that accounts for potential alignment discrepancies, we attain a 75% success rate in completely recovering the value of $\mathbf{cs}_1$.

- Given that z is known in $z = y + \mathbf{cs}_1$, we utilize the relatively more precise HW information from side-channel results for y to augment the success rate of $\mathbf{cs}_1$ recovery. By amalgamating the attack outcomes on y with those on $\mathbf{cs}_1$, we elevate our recovery success rate of $\mathbf{cs}_1$ to 92%, marking a significant leap in attack efficiency.

- To tackle the issue of erroneous equations in the recovery of $\mathbf{cs}_1$, we introduce a constrained optimization-based residual analysis. This innovative approach rapidly solves the system of integer linear equations laden with errors, leveraging the constraints inherent in the private key $\mathbf{s}_1$. Notably, this technique proves effective even with only 10% accuracy in the system of equations, provided that a sufficient number of equations are present. This substantially narrows down the private key value space and expedites the ILP process.

- Our practical assault on the Dilithium2 reference implementation on an STM32F4 platform is demonstrated across three scenarios: access to only y, only $\mathbf{cs}_1$, and both y and $\mathbf{cs}_1$. The outcomes in the worst-case scenarios illustrate that targeting y enables private key recovery within 2 minutes using 8 signatures; attacking $\mathbf{cs}_1$ necessitates 3 signatures; and harnessing leaks from both y and $\mathbf{cs}_1$ facilitates private key recovery in 3 minutes with merely 2 signatures. In these tests, a 60% probability of recovering the private key with a single signature was observed.

2 Preliminaries

2.1 Dilithium

Dilithium is a digital signature scheme based on the principles of the Module Learning with Errors (MLWE) and Module Short Integer Solution (MSIS) problems. Its design allows for different security levels, making it suitable for a wide range of uses. This adaptability ensures that Dilithium can match the varied security and performance needs of different devices. Tab.1 delineates the parameter configurations for each security level.

Table 1: Dilithium parameters at different NIST security levels

NIST Security Level	2	3	5
d [dropped bits from $\boldsymbol{t}$]	13		
τ[# of non-zero coefficients in $\boldsymbol{c}$]	39	49	60
γ_1 [cofficient range of $\boldsymbol{y}$]	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 process, 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: $\mathbf{y} \in S_{\gamma_1 - 1}^n := \text{ExpandMask}(\rho', \kappa)$
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: $\mathbf{r}_0 := \text{LowBits}_q(\mathbf{w} - \mathbf{c}\mathbf{s}_2, 2\gamma_2)$
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:     $\mathbf{h} := \text{MakeHint}_q(-\mathbf{c}\mathbf{t}_0, \mathbf{w} - \mathbf{c}\mathbf{s}_2 + \mathbf{c}\mathbf{t}_0, 2\gamma_2)$
16:     if $||\mathbf{c}\mathbf{t}_0||_\infty \geqslant \gamma_2$ or the # of $1^{'}$s in $\mathbf{h}$ is greater than $\omega$
            then $\kappa := \kappa + l$, goto 6
17: return $signature = (\mathbf{z}, \mathbf{h}, \tilde{\mathbf{c}})$
```

The signature process of Dilithium, as outlined in Alg.1, commences with the input of a secret key sk and a message M. Initially, the algorithm expands the secret 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 Cryptographic Hash Function (CRH). 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 deterministic or randomised schemes. Subsequently, the NTT is applied to both $\mathbf{A}$ and the secret $\mathbf{s}_1$, generating $\hat{\mathbf{A}}$ and $\hat{\mathbf{s}}_1$. A masking vector$y$ is derived from ρ' and κ, within the set S_{γ_1-1}. The algorithm computes w by multiplying $\hat{\mathbf{A}}$ with the NTT of y and applying the Inverse NTT (INTT), then extracts high-order bits from w to form a challenge vector $\tilde{\mathbf{c}}$, integral to the signature's validity. The algorithm features a rejection sampling loop to ensure the generated vectors $\mathbf{z}$ and $\mathbf{r}_0$ meet specific security criteria. If the criteria are not met, the algorithm recalibrates $\mathbf{y}$ and iterates again, ensuring compliance with Dilithium's security standards. Upon meeting these standards, the algorithm produces a signature comprising $\mathbf{z}$, a hint vector $\mathbf{h}$ for verification, and $\tilde{\mathbf{c}}$.

2.2 Linear Regression-based Profiled Attacks

Schindler *et al.* [SLP05] introduced Linear Regression-based (LR) profiled attacks, marking a significant shift from the Hamming Weight (HW) leakage model in traditional template attacks [CRR02]. They take into account that different bits might have different leakage weights, a nuance that linear regression can accurately identify.

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, with y_i being the i-th bit from least to most significant, and l_y denotes the length of y in bits. The coefficients a_i indicate the leakage weight of each bit, while $\rho(y_i)$ is a mapping function that ensures the influence of $y_i = 0$ is considered in the model. This mapping is specifically chosen to accurately reflect the contribution of each bit value to the model.

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

In the phase of building templates, real power leakages, L, are used to calculate the coefficients a_i through the linear least squares approach. The model $m(y)$ then serves as the mean in traditional template attacks, but with an additional step required to compute the covariance matrix Σ for generating the templates. During the attack phase, for any observed power leakage L, the probability density function 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 probability for each y, simplifying the expression 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 SCA is predicated on the assembly of a comprehensive training dataset, comprising leakage traces, plaintext-ciphertext pairings, and corresponding cryptographic keys. Each trace is meticulously labeled with sensitive intermediate values, thus categorizing the data and furnishing the CNN with supervisory signals for the duration of the training phase.

CNNs are characterized by their intricate structure, which includes convolutional operations, setting them apart in the realm of side-channel analysis. Their effectiveness is attributed to a 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]:

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

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 power traces. Once trained, the CNN can effectively predict sensitive intermediate values from previously unseen traces and ultimately recover sensitive information.

3 Side-channel Attacks Against Dilithium

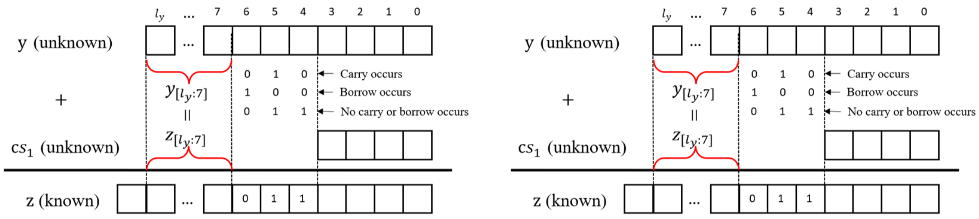
In addressing the polynomial addition process, represented as $z = y + \mathbf{cs}_1$, the known signature z offers an indirect pathway to deduce $\mathbf{cs}_1$ by initially recovering y through a side-channel attack. Alternatively, $\mathbf{cs}_1$ can be directly recovered via side-channel analysis. By merging insights gained from the side-channel attack on y with those from the attack on $\mathbf{cs}_1$, we expect a significant boost in the effectiveness of our cryptographic analysis. 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 y, operating within the value range of $q = 8380417$, and considering that each signature y is randomly generated, we are restricted to utilizing only one trace for y's recovery in a practical attack scenario. The direct implementation of a conventional template attack, using the identity model, is theoretically possible but practically challenging due to the need for a large dataset for accurate templates building and the associated low success rate during template matching.

In practical scenarios, particularly 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, β's largest allowable value—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 improves the efficiency of the attack.

Liu *et al.* [LZS+21, QLZ+23a] have highlighted the methodology for inferring the high-bit information of y when access to 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 $z_{[i-1:\tau]} = 10\dots00_2$ or $z_{[i-1:\tau]} = 01\dots11_2$, 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 streamlined method for creating models, necessitating fewer training samples to discern the complete target value, not just its HW. 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_i values, comparing the theoretical leakage predicted by the LR model against actual trace observations. Crucially, bits within varying y_i 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 within the Dilithium cryptographic scheme 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 developed to represent this leakage data accurately.

(2) Trace 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 actual observed 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.

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

Using Dilithium2 as an example, we find that about 95% of the data falls within $||\boldsymbol{cs}_1||_\infty < 2^4$, with the remainder 5%, as detailed in Tab.2, covering a broader value range of $(\pm 16, \pm 78)$. These outliers encompass a significantly larger value space but occur with much less frequency. Including all these potential values in the attack might paradoxically decrease the success rate due to the dilution of focus on the most probable scenarios. By intentionally excluding these less likely outliers from the analysis, the success rate and efficiency of the attack can be significantly enhanced, focusing efforts where they are most likely to yield results.

3.2 Attacking $\mathbf{cs}_1$ with CNN-Based Side-channel Attack

In addition to focusing on the random number y in side-channel attacks, directly targeting the $\mathbf{cs}_1$ value emerges as a viable alternative. Due to its inherently smaller value space, $\mathbf{cs}_1$ naturally presents more advantageous conditions for successful exploitation. The attack scenario is similar to y in that only where only one $\mathbf{cs}_1$ component can be recovered from a single trace. This highlights the crucial importance of template quality in influencing the success rate 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 × 4 × 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, introducing offsets as an inevitable byproduct. Deep learning approaches, particularly CNN, have demonstrated exceptional proficiency 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 $N^{[0]} = 50$ and 270 for $N^{[0]} = 100$.

Drawing inspiration from Zaid *et al.*'s methodology, 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.

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`
Metric	Accuracy

Leveraging a similar approach, our model excels in identifying and learning intricate patterns present in collected traces. Its architectural design facilitates a progressive extraction and condensation of features, ensuring an efficient representation of pertinent information. This capability not only enhances robust classification performance but also preserves computational efficiency.

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 power 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 direct attack does not succeed outright, the probability distribution vectors related to the attack target often highlight a higher likelihood for candidate values matching the correct result's HW. 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 Resolving Equations with Errors in $\mathbf{cs}_1$ for Dilithium

Extracting $\mathbf{cs}_1$ via side-channel or fault attacks precedes solving for $\mathbf{s}_1$ using the known challenge ciphertext $\mathbf{c}$. Achieving perfect accuracy in determining $\mathbf{cs}_1$ is inherently challenging, regardless of the technique. 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-laden 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 $\mathbf{cs}_1$ 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 computational steps 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 challenge ciphertext $\mathbf{c}$ is characterized by specific values that negate the need for modular operations in its computation. 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 $\mathbf{c}$, $\mathbf{s}$ 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 side-channel analysis. Assuming an attacker achieves a 30% success rate in acquiring $\mathbf{b}$, the collection of ten signatures yields 2560 equations for $\mathbf{s}$, with 768 being accurate. 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, isolating 256 correct equations from the accurate subset facilitates the recovery of the actual private key by solving a 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 main phases. The first consists of transforming the problem into a continuous optimization framework. Here, the attacker aims to reduce the overall sum of squared residuals for all equations, while temporarily disregarding the integer constraints on $\mathbf{s}$. This objective is mathematically expressed as:

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

The initial phase of filtering erroneous equations is 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 specifically for such challenges. 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 the optimization procedure, the subsequent phase entails the evaluation of residuals for each equation within 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 $r_i = |\mathbf{A}_i\mathbf{s}^* - \mathbf{b}_i|$, 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 rationale behind this exclusion process is that removing equations with significant discrepancies from the current solution can lead to a more accurate approximation of the true solution by reducing the influence of potential errors.

The iterative refinement process recalculates the solution $\mathbf{s}^{(k+1)}$ at each iteration k, using the updated equation set $\mathbf{A}^{(k+1)}$ and $\mathbf{b}^{(k+1)}$ that omit previously identified incorrect equations. This refinement is mathematically formulated as follows:

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

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 $\mathbf{b}$ in the updated equation set.

To finalize the iterative optimization, 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, $\mathbf{s}_{\text{int},i}$,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 from the original set. Subsequent verification of the integer solution is advised to confirm its adequacy in satisfying the linear system to an acceptable degree.

Based on this principle, we introduce the Constrained Optimization-Based Residual Analysis (COBRA) method. This approach efficiently isolates correct equations from a dataset significantly contaminated with inaccuracies, thereby facilitating the accurate derivation of the private key. The process, outlined in the pseudo-code of Alg.2, operates with inputs such as the coefficient matrix $\mathbf{C}$ derived from the challenge $\mathbf{c}$, results from

Algorithm 2 Constrained Optimization-Based Residual Analysis

```
Input: $\mathbf{C}, \boldsymbol{sca_r}, bounds, e_{th}, d_{num}, r_{num}, s_{num}$
Output: $\mathbf{s}$
$available\_ind \leftarrow$ InitializeIndices($\mathbf{C}$)
$err\_weights \leftarrow \mathbf{zeros}(|\mathbf{C}|)$
while $|available\_ind| > r_{num}$ do
    $s\_ind \leftarrow$ RandomSample($available\_ind, s_{num}$)
    $\mathbf{s} \leftarrow$ SolveLSQ($\mathbf{C}[s\_ind], \boldsymbol{sca_r}[s\_ind], bounds$)
    $residuals \leftarrow$ CalculateResiduals($\mathbf{C}[s\_ind], \boldsymbol{sca_r}[s\_ind], \mathbf{s}_1$)
    $err\_weights \leftarrow$ UpdateWeights($residuals, err\_weights, d_{num}$)
    $available\_ind \leftarrow$ UpdateIndices($err\_weights, e_{th}$)
end while
if $\mathbf{C}[available\_ind]\mathbf{s} == \boldsymbol{sca_r}[available\_ind]$ then
    return $\mathbf{s}$
end if
```

the side-channel attack denoted as sca_r, solution *bounds*, error threshold e_{th}, a delta for incrementing error weights each iteration d_{num}, a retention threshold for the least erroneous 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 side-channel 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 based on the magnitude of these residuals. 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 refined set of equations aligns perfectly with the solution, evidenced by zero residuals, thus indicating successful error mitigation and recovery of the correct solution.

Critical to the algorithm's success are parameters like the error threshold e_{th} and the iterative incrementation of 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 correct recovery might otherwise be infeasible. 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, resolving 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 by accurately identifying a subset of correct equations. This observation enables the application of ILP to a reduced set of 200 equations for Dilithium2. Armed with this knowledge, an attacker could feasibly achieve full private key recovery utilizing merely a single signature.

4.2 Big-M with Constrained Optimization-Based Residual Analysis

The challenge at hand is to identify 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, the coefficients it retrieves often include values that are nearly correct, with many incorrect coefficients deviating by 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.

The algorithm, referred to as Alg.2, introduces randomness by selecting indices randomly in each iteration, which results in varied outcomes. If a single run of Alg.2 does not recover the complete private key, repeating its core computational module might yield alternative solutions. 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.

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

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

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.

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 resolving integer linear equations related to $\mathbf{cs}_1$,drawing from side-channel attack outcomes. The BP algorithm iteratively refines solution estimates by updating and circulating local marginal probabilities within a factor graph, with side-channel data integration enhancing solution precision. 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, while $\mathbf{s}$ spans a uniform distribution over $\{-\eta, \ldots, \eta\}^n$.

The BP algorithm needs to rely on more equations when the success rate of the side channel 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. One natural strategy for recovering the entire private key with known partial coefficients is reducing it to a special LWE problem. When the positions of recovered coefficients are known, it can be regarded as a form of leaky LWE problem and is resolved utilizing the leaky LWE estimator proposed by Dachman-Soled *et al.* [DDGR20]. In this case, 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 $\mathbf{c}\mathbf{s}_1 = \mathbf{b}$, 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 $\mathbf{c}\mathbf{s}_1 = \mathbf{b}$ and substituting it into the public key $\mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$, we can obtain a new LWE problem $\mathbf{t}' = \mathbf{A}\mathbf{s}_1' + \mathbf{s}_2$ where $\mathbf{t}' = \mathbf{t} - \mathbf{A}\mathbf{s}_1^*$, $\mathbf{s}_1' = \mathbf{s}_1 - \mathbf{s}_1^*$. If most equations in $\mathbf{c}\mathbf{s}_1 = \mathbf{b}$ 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 of the Dilithium algorithm, as per the NIST reference [ACD+22]. To capture leakage signals, 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 on power leakages associated with the operation of generating random numbers y in the Dilithium. Their research identifies

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 $\mathbf{t}' = \mathbf{A}\mathbf{s}_1' + \mathbf{e}$ where $\mathbf{e} = \mathbf{s}_2 - \mathbf{t}_0$, $\mathbf{t}_0$ denotes the low order bits of $\mathbf{t}$.

the "polyz_unpack(a, buf)" function as exhibiting the most substantial leakage related to y, thereby designating this function as the target of our attack. The function, as detailed in Fig.3, is responsible for extracting an 18-bit value from a predefined random array, denoted as r, and converting it into y via the operation GAMMA1-r, where GAMMA1 $= 2^{17}$. This conversion is executed over 64 cycles, yielding four instances of y per cycle.

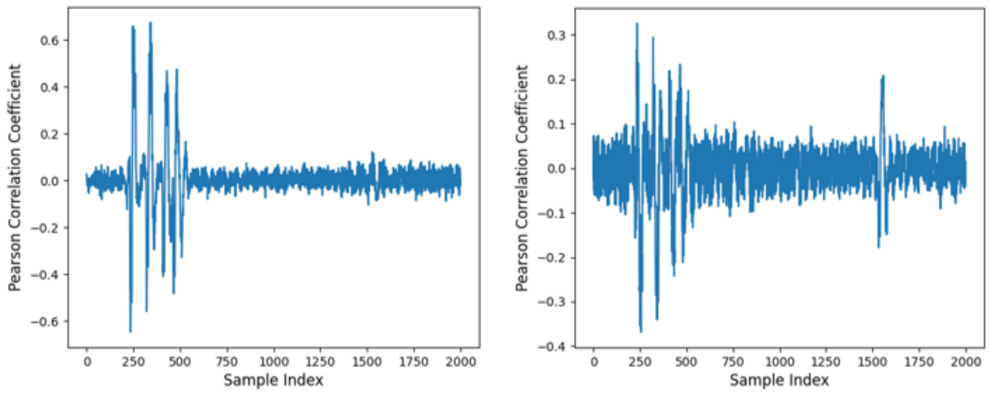
```c
void polyz_unpack(poly *r, const uint8_t *a) {
    unsigned int i;

#if GAMMA1 == (1 << 17)
    for(i = 0; i < N/4; ++i) {
        r->coeffs[4*i+0] = (uint32_t)a[9*i+0] << 8;
        r->coeffs[4*i+0] |= (uint32_t)a[9*i+1];
        r->coeffs[4*i+0] &= 0x3FFFF;

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

        r->coeffs[4*i+2] = a[9*i+3] >> 4;
        r->coeffs[4*i+2] |= (uint32_t)a[9*i+4] << 4;
        r->coeffs[4*i+2] |= (uint32_t)a[9*i+5] << 12;
        r->coeffs[4*i+2] &= 0x3FFFF;

        r->coeffs[4*i+3] = a[9*i+5] >> 6;
        r->coeffs[4*i+3] |= (uint32_t)a[9*i+6] << 2;
        r->coeffs[4*i+3] |= (uint32_t)a[9*i+7] << 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];
    }
#endif}
```

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

Our goal is to recover y through side-channel analysis, emphasizing the leakage of its lower bits as outlined in 3.1. However, our correlation analysis of 1,000 traces using the HW model, as shown in Fig.4, indicates that focusing on r, especially its lower 8 bits, results in more significant leakage. This finding is attributed to the direct computational correlation between r and y, as well as the additional operations on r. Therefore, attacking the operation corresponding to r will have better results significantly higher than attacking y directly.

For the Dilithium2 scheme, we employ 50,000 traces and apply a low-pass filter using FFT to improve signal quality. Subsequently, we identify continuous points of interests(POIs) that exhibit significant leakage for modeling through a LR model. This approach ensures that y is modeled independently for each coefficient position, which helps to mitigate potential biases arising from trace misalignment. After building the templates, the attack is carried out on traces corresponding to the new signature under analysis.

In our experiments, we methodically investigate how varying the number of POIs of our attack, focusing two scenarios based on distribution of $\mathbf{cs}_1$: either $||\mathbf{cs_1}||_\infty < 2^4$ or $||\mathbf{cs_1}||_\infty < 2^5$. The findings, detailed in Tab.4 for 256 coefficients profiled and attacked, reveal the relationship among the number of POIs, the profiling time, and the attack's success rate (SR). Notably, with 20 POIs, the profiling time amounts to 134.2 seconds, leading a 14.2% SR under the 4-bit assumption within 7.2 seconds, and an 8.8% SR under



(a) PCC of r

(b) PCC of y

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

the 5-bit assumption in 15.89 seconds. When the POIs increase to 300, the profiling time rises to 952.3 seconds, but this also significantly enhances the SR to 39.6% for the 4-bit assumption at 240.6 seconds, and to 33.4% for the 5-bit assumption at 461.3 seconds.

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 $\mathbf{cs}_1$ within the Dilithium2 framework. The distribution of $\mathbf{cs}_1$ closely approximates a Gaussian curve, with approximately 95% of the data falling within a 4-bit range. This Gaussian-like distribution indicates that under the 4-bit assumption, the attack already encompasses the vast majority of possibilities, thereby reducing the enumeration likelihood and consequently enhancing the attack's success rate. On the other hand, 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 it might appear that this extension allows for the recovery of a broader range of values, it actually diminishes the attack's effectiveness. This paradox underscores the pivotal role of $\mathbf{cs}_1$'s Gaussian-like distribution in dictating the attack's success rates.

Our method demonstrates the capability to fully recover the value of y with success rates nearing 40% using single trace, representing a significant advancement in efficiency compared to previous approaches.This improvement in efficiency is primarily attributed to the reduced need for multiple traces to build templates and a smaller enumeration space during template matching. Importantly, our attack strategy obviates the requirement for selecting or filtering y values, thereby enhancing its efficiency further. Additionally, the independence of y's coefficient positions enables the concurrent execution of attacks on these coefficients. This approach not only expedites the attack process but also underscores the versatility and practical applicability of our technique in the realm of side-channel attacks.

5.3 DL-SCA for Recovering $\mathbf{cs_1}$

Chen *et al.* [CKA+21] identified significant leakage in the Montgomery reduction operation, a pivotal element of the Dilithium algorithm's reference implementation submitted to NIST. This operation, crucial to the entire INTT process, facilitates the conversion of results from the NTT domain back to the normal domain for further calculations. The specific implementation of the Montgomery reduction is illustrated in Fig.5. Notably, the reduction process, particularly the shifting and storage operations in its final stages, is identified as a likely primary source of significant $\mathbf{cs}_1$ leakage. Our targeted experimental attacks on these operations revealed a maximum SNR of 2.5 for the leakage related to $\mathbf{cs}_1$.

```
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 has deterministic and randomised schemes. The deterministic scheme enables the recovery of all discarded challenge $\mathbf{c}$ values when the private key is known, which is not possible with the randomized scheme. In our analysis, we specifically focus on the leakages stemming from the final round of legitimate signatures. For Dilithium2, our data collection is concentrated on capturing the complete round of $\mathbf{cs}_1$ leakage. While it's theoretically feasible to collect a broader dataset by including $\mathbf{cs}_2$ leakage, our current discussion remains focused solely on $\mathbf{cs}_1$, excluding $\mathbf{cs}_2$ from our scope.

For Dilithium2, each signature provided 1,024 actionable training samples. We statically align and segment these samples into blocks of 400 for training the CNN model, as detailed in Sec.3.2. The training strategy addresses the distribution skewness of $\mathbf{cs}_1$, by focusing on data within $||\mathbf{cs_1}||_\infty < 2^4$ and $||\mathbf{cs_1}||_\infty < 2^5$, as well as considering the full data range. This comprehensive approach involves 250,000 samples and is completed about 40 minutes.

Fig.6 illustrates the guess entropy and success rates for side-channel attacks under various strategies. With consistent model hyperparameters, expanding the training set enables the creation of more accurate classifiers. Utilizing 10,240 samples (equivalent to 10 signatures) leads to a stabilization in both the classifier's success rate and guess entropy. Generally, when $\mathbf{cs}_1$ is assumed to be 5 bits, the outcomes marginally surpass those for the 4-bit assumption, with success rates improving from 70% to 74%. For guess entropy, the assumption of $\mathbf{cs}_1$ as 5 bits approaches 1. The primary reason for similar success rates yet significant differences in guess entropy is due to the fact that, under the assumption of $\mathbf{cs}_1$ being 4 bits, certain actual values fall outside our hypothesized space, and these cases are ranked last in our analysis. These findings underscore the potential of CNN-based techniques, suggesting that further optimization of network hyperparameters might lead to enhanced success rates.

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, in addition to attempts at using all available points. The success rates, as detailed in Tab.5, demonstrate a trend where an increased number of POIs generally boosts the attack's effectiveness. Notably, 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.

Despite the incremental gains observed in TAs with an expanded selection of POIs, CNN-based attacks have consistently surpassed TAs in terms of both success rates and computational efficiency. This highlights the advantages of CNNs in improving the success

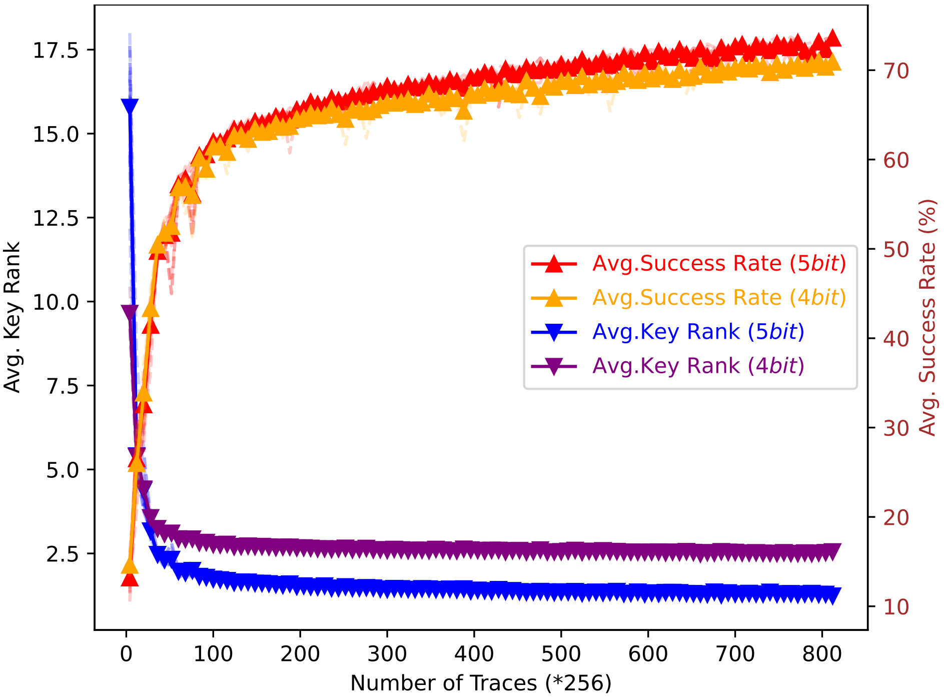


Figure 6: Guess entropy and success rates of $\mathbf{cs}_1$

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 ALL)	54.6	57.7	57.5

and efficiency of side-channel attacks, especially when utilizing all feature points. Therefore, CNNs emerge as the preferred method for analyzing and exploiting $\mathbf{cs}_1$ leakage in the Montgomery reduction process.

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 employed the method described in Sec.3.3 to develop a more potent attack by combining the results for y with those for $\mathbf{cs}_1$. Tab.6 presents the success rates achieved, showcasing the impact of altering the PoIs for y. With the assumption that $||\mathbf{cs_1}||_\infty < 2^4$, we achieve an optimal success rate of 86%. This rate further increases to an average of 92.8% when the model presupposes that $||\mathbf{cs_1}||_\infty < 2^5$. These results compellingly demonstrate the enhanced efficiency of the combined attack strategy, significantly improving its effectiveness.

5.4 Constrained Optimization-Based Residual Analysis Result

In our empirical investigation, we found that when the set of 256 equations includes more than 8 inaccuracies, retrieving the private key becomes impractical within an hour using the Big-M method on our computational setup. We utilized simulation data to assess the effectiveness of constrained optimization-based residual analysis across different success rates, with the goal of setting benchmarks for future practical attacks.

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

Table 7: Performance of Constrained Optimization-Based Residual Analysis

Correct Eq. Ratio	Eq. Count	$\mathbf{d_{num}}$	$\mathbf{e_{th}}$	SR (%)	Time (s)
10%	65 × 256	50	0	100	189.5
30%	9 × 256	50	10	100	53
50%	4 × 256	50	5	100	10.0
70%	3 × 256	40	0	100	1.63
90%	2 × 256	30	0	100	0.8
95%	1 × 256	2	2	83	4.1

Tab.7 presents the necessary equations and specific parameter adjustments for successful private key recovery, given varying percentages of correct equations. It shows that achieving private key recovery is feasible across different success rates, provided a certain threshold number of equations is met. Generally, the need for equations decreases as the success rate increases. This parameterization is heuristic, designed for flexibility rather than pushing theoretical limits, allowing for real-time adjustments to improve practical implementation efficiency. As the proportion of incorrect equations increases within the set, it becomes imperative to increment d_{num} to expedite the removal of these inaccuracies. While a minimum of 230 correct equations, denoted as r_{num} , can potentially facilitate private key recovery in specific scenarios, adjusting parameters based on real-world conditions can enhance the speed of the attack.

Particularly in scenarios where only 10% of the equations are correct, successful attacks have been executed with fewer than 60 × 256 equations (corresponding to 60 signatures), albeit at the cost of significant time overhead due to increased e_{th}. If attackers have access to a large set of equations, increasing the number of equations is recommended to quicken the resolution process. However, setting d_{num} too high should be avoided, as it may mistakenly eliminate a significant number of correct equations early on.

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.

Moreover, if attackers are dealing with a lower number of equations, our technique still significantly reduces the entropy of the private key and decreases the time complexity associated with solving ILP. This benefit aids in the execution of the actual attack, highlighting our method's utility and efficiency in the field of cryptographic analysis.

5.5 Practical SCAs of Dilithium

The practical attack on Dilithium2 provided enlightening findings. Considering that attackers in real-world scenarios may only access leakage from y and $\mathbf{cs}_1$ separately, we outline the detailed attack results for three different scenarios: targeting y, $\mathbf{cs}_1$, and a combined approach where leakage from both y and $\mathbf{cs}_1$ is utilized simultaneously.

Table 8: 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 optimal case that occurs in the attack.

Tab.8 documents the number of signatures needed for private key recovery under different scenarios, showcasing both the highest and lowest number of signatures required in successful attempts among a series of 10 experiments. Interestingly, the efficiency of the attack shows minimal fluctuation with different assumptions about the bit range of $\mathbf{cs}_1$. Targeting the leakage of y alone requires around 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, and in cases assuming $||\mathbf{cs_1}||_\infty < 2^5$, often only 2 signatures are needed. Exploiting leakage from both y and $\mathbf{cs}_1$ require at most 2 signatures to recover $\mathbf{s}_1$. Notably, our empirical analysis revealed that about 15% of cases could succeed with leakage from a single signature, usually requiring the recovery of 240 out of 256 coefficients through the side-channel attack.

These outcomes underscore the robustness of our approach, demonstrating that even when attacking solely on y at a modest success rate, the attack can be executed with fewer than 10 signatures—an advancement over preceding efforts by Marzougui *et al.* [MUTS22] and Berzati *et al.* [BVC+23].

Furthermore, our refined attack strategy, integrating a hybrid approach with the voting technique and ILP as detailed in Sec.4.2, underwent 1,000 repetitions under the Constrained Optimization-Based Residual Analysis algorithm to refine the potential values for $\mathbf{s}_1$, typically concluding within 20 minutes. This preparatory phase was succeeded by leveraging an optimized Big-M method for final solution derivation. Observations revealed that when the side-channel attack successfully identifies more than 236 coefficients, the ILP phase generally concludes within 30 minutes. Should the recovered coefficients fall below this threshold, solution times often extend beyond one hour, vastly surpassing the original scheme's tolerance for up to 8 errors within a similar timeframe, hence marking a considerable enhancement. In this experimental framework, approximately 60% of datasets facilitated the recovery of the private key within an hour using a singular signature.

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 $\mathbf{cs}_1$, the challenge remains that even with 230 coefficients recovered and after reducing the constraints with our constrained optimization-based residual analysis, the computational expense of solving through ILP remains significantly high. We posit that amalgamating our side-channel findings with the BP algorithm could facilitate a more consistent realization of the 1-signature attack. In future endeavors, we aim to explore more efficient mathematical methods to achieve Dilithium attacks under single signatures with improved stability and efficiency. Furthermore, we plan to investigate the effectiveness of our approach against protected implementations of Dilithium, potentially offering insights into enhancing the security measures against side-channel attacks.

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.

[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.