An Efficient Non-Profiled Side-Channel Attack on the CRYSTALS-Dilithium Post-Quantum Signature

Zhaohui Chen
School of Computer Science and Technology
University of Chinese Academy of Sciences
Beijing, China
chenzhaohui17@mails.ucas.ac.cn

Emre Karabulut
Department of Electrical and Computer Engineering
North Carolina State University
NC, USA
ekarabu@ncsu.edu

Aydin Aysu
Department of Electrical and Computer Engineering
North Carolina State University
NC, USA
aaysu@ncsu.edu

Yuan Ma†
State Key Laboratory of Information Security
Institute of Information Engineering, CAS
Beijing, China
mayuan@iie.ac.cn

Jiwu Jing
School of Cryptography
University of Chinese Academy of Sciences
Beijing, China
jwjing@ucas.ac.cn

***Abstract*—Post-quantum digital signature is a critical primitive of computer security in the era of quantum hegemony. As a finalist of the post-quantum cryptography standardization process, the theoretical security of the CRYSTALS-Dilithium (Dilithium) signature scheme has been quantified to withstand classical and quantum cryptanalysis. However, there is an inherent power side-channel information leakage in its implementation instance due to the physical characteristics of hardware.**

This work proposes an efficient non-profiled Correlation Power Analysis (CPA) strategy on Dilithium to recover the secret key by targeting the underlying polynomial multiplication arithmetic. We first develop a conservative scheme with a reduced key guess space, which can extract a secret key coefficient with a 99.99% confidence using 157 power traces of the reference Dilithium implementation. However, this scheme suffers from the computational overhead caused by the large modulus in Dilithium signature. To further accelerate the CPA run-time, we propose a fast two-stage scheme that selects a smaller search space and then resolves false positives. We finally construct a hybrid scheme that combines the advantages of both schemes. Real-world experiment on the power measurement data shows that our hybrid scheme improves the attack's execution time by 7.77×.

***Index Terms*—Hardware Security, Post-quantum Cryptography, Correlation Power Analysis, Digital Signature, Number Theoretic Transform**

I. Introduction

The digital signature (DS) algorithms are widely used in protecting computer security regarding integrity and non-repudiation. In many scenes such as authenticating Trusted Platform Module (TPM) and firmware update, the DS is used as a basic cryptography primitive. The traditional DS algorithms are based on the difficulty of large integer factorization [1] or the discrete logarithm problem [2]. However, the quantum algorithm [3] can solve these problems exponentially faster than the best currently-known algorithm running on a classical computer.

In December 2016, the National Institute of Standards and Technology of the United States announced the standardization process of public-key cryptography algorithms against quantum computer attacks, including public-key encryption (PKE), key encapsulation mechanism (KEM), and DS algorithms. After three rounds of competition, there remain three finalists in the DS area. DS finalist CRYSTALS-Dilithium (Dilithium) and a promising PKE/KEM finalist scheme named CRYSTALS-Kyber (Kyber) consist of the CRYSTAL cipher suite. Because the two algorithms have similar arithmetic structures, CRYSTAL has advantages in module reuse and technology diffusion.

The Dilithium post-quantum DS algorithm faces both mathematical attacks and physical attacks. For the former aspect, the authors provide security analysis against Block–Korkine–Zolotarev (BKZ) algorithm [4], algebraic attacks [5], and dense sub-lattice attack [6], among others, based on quantum or classical computers. Dilithium, moreover, guarantees Strong Existential Unforgeability under Chosen Message Attack (SUF-CMA). Physical attacks include secret information extraction from side-channel leakage such as execution time, power consumption, and electromagnetic radiation. These attacks lead to existential forgery attacks or key recovery attacks, both of which are disastrous for a computer system.

The existing work has confirmed the side-channel leakage risk of Dilithium. Migliore *et al.* [7] test several operations with the classical Welch's T-test. They claim a generic leakage but did not provide a specific scheme to break Dilithium. Existing profiled attacks on forward Number Theoretic Transform (NTT) cannot extend to Dilithium [8] because these operations could be pre-computed. Moreover, such profiled

†This author is the corresponding author. He is also with School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China.

attacks require the adversary to have access to an equivalent device and reconfiguration capability to build templates [9], [10]. By contrast, non-profiled attacks such as the Differential Power Analysis (DPA) [11] and Correlation Power Analysis (CPA) [12] do not have this limitation.

Fournaris *et al.* [13] argues that the straightforward DPA attack on coefficient-wise multiplication is feasible, but they again do not provide the attack implementation or any attack results. There is also side-channel assisted mathematical attack [14] where the randomness leakage in polynomial addition results in a security reduction, which allows polynomial-time key recovery. However, this attack is limited to Fiat-Shamir signatures that use a specific formula in signature generation. Likewise, other attacks targeting schoolbook or efficient sparse polynomial multiplier [15]–[17] are not directly applicable to Dilithium because it use NTT-based multiplication.

An efficient CPA scheme can be destructive in the real-world applications of post-quantum cryptosystems. However, the direct non-profiled attack on Dilithium's implementation has not been explored in depth. Our analysis has revealed several key challenges and found novel ways to address them. The contributions of this work are listed as follows:

- We analyze the power leakage by targeting polynomial point-wise multiplication and transfer the classical CPA to this Point of Interest (PoI). We reveal that the large modulus and implementation of Dilithium results in a large side-channel search space of 2^{27}. By minimizing the key guess space with an algebraic analysis of the correct guesses, our conservative CPA scheme reduces the computational overhead by $32\times$.
- We propose a fast two-stage scheme to further accelerate the attack's run-time. We find a PoI which only depends on a segment of the secret key although generating false positives. The first stage uses this imperfect PoI to eliminate wrong key guesses while recording several candidates of the key segment. The second stage uses an ideal PoI to filter out those false positives and recover the complete key coefficient.
- We develop a hybrid scheme that collocates both the conservative scheme and the fast scheme to combine their benefits. This attack gives priority to the fast scheme if power traces are sufficient. The adversary can judge whether the fast scheme hits or not by comparing the peak value with a threshold. If it hits, the adversary can directly output the correct key. Otherwise, the adversary can use the conservative scheme as a backup.
- Our experiments on an off-the-shelf ARM Cortex-M4 processor show that the adversary can recover the key coefficient with a 99.99% confidence. The proposed hybrid scheme achieves $7.77\times$ acceleration, which saves about 3403 compute hours over the conservative scheme on recovering the key of Dilithium-II.

Algorithm 1: Framework of Dilithium signature (Key Generation, Signing and Verification)

KeyGen
$\mathbf{A} \leftarrow R_q^{k\times\ell}$
$(\mathbf{s}_1, \mathbf{s}_2) \leftarrow S_\eta^\ell \times S_\eta^k$
$\mathbf{t} := \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2$
$(\mathbf{t}_1, \mathbf{t}_0) := \text{Power2Round}_q(\mathbf{t}, d)$
return $(pk = (\mathbf{A}, \mathbf{t}_1), sk = (\mathbf{A}, \mathbf{t}_0, \mathbf{s}_1, \mathbf{s}_2))$
Sign(sk, M)
$(\mathbf{z}, \mathbf{h}) := \perp$
while $(\mathbf{z}, \mathbf{h}) = \perp$ **do**
  $\mathbf{y} \leftarrow S_{\gamma_1}^\ell$
  $\mathbf{w} := \mathbf{A}\mathbf{y}$
  $\mathbf{w}_1 := \text{HighBits}(\mathbf{w}, 2\gamma_2)$
  $c \in B_\tau := \text{H}(M \| \mathbf{w}_1)$
  $\mathbf{z} := \mathbf{y} + c\mathbf{s}_1$
  if $\|\mathbf{z}\|_\infty \geq \gamma_1 - \beta$ or $\| \text{LowBits}(\mathbf{w} - c\mathbf{s}_2, 2\gamma_2) \|_\infty \geq \gamma_2 - \beta$ **then**
    $(\mathbf{z}, \mathbf{h}) := \perp$
  end
  else
    $\mathbf{h} := \text{MakeHint}_q(-c\mathbf{t}_0, \mathbf{w} - c\mathbf{s}_2 + c\mathbf{t}_0, 2\gamma_2)$
    if $\|c\mathbf{t}_0\|_\infty \geq \gamma_2$ or No. '1' in $\mathbf{h} \geq w$ **then**
      $(\mathbf{z}, \mathbf{h}) := \perp$
    end
  end
end
return $\sigma = (\mathbf{z}, \mathbf{h}, c)$
Verify$(pk, M, \sigma = (\mathbf{z}, \mathbf{h}, c))$
$\mathbf{w}_1' := \text{UseHint}_q(\mathbf{h}, \mathbf{A}\mathbf{z} - c\mathbf{t}_1 \cdot 2^d, 2\gamma_2)$
return $[\![\|\mathbf{z}\|_\infty < \gamma_1 - \beta]\!]$ and $[\![c = \text{H}(M \| \mathbf{w}_1')]\!]$ and $[\![\text{No. '1' in } \mathbf{h} \leq \omega]\!]$

II. Preliminaries

This section introduces the Dilithium DS and its parameter settings. This part focuses on the operations involving the secret key, which adversaries are interested in. This section further introduces the capabilities of adversaries and the general CPA methods to recover the secret keys.

A. Notations

The matrix is represented by bold capital letters, such as a matrix $\mathbf{A}$. Vectors are represented by bold lowercase letters, such as a vector $\mathbf{s}$. In the Dilithium DS, the entries in the underlying matrix or vector belong to the polynomial ring $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ by default. Lowercase italics represent polynomials or integer parameters. A polynomial belonging to R_q is of order $n - 1$, with integer coefficients modulo a prime number q.

Further, multiplication of two polynomial structures $a, b \in R_q$ is denoted as $ab \in R_q$. Point-wise multiplication of two polynomials a and $b \in R_q$ is denoted as $a \circ b \in R_q$. Integer multiplication of a and b is denoted as $a \cdot b$. For an element $w \in \mathbb{Z}_q$, $\|z\|_\infty$ means $|z \bmod^{\pm} q|$, while $\|\mathbf{z}\|_\infty$ represents the maximum value of this operator among all coefficients in the vector $\mathbf{z}$. Formula $\mathbf{s} \leftarrow S_\eta^k$ means the k-dimensional polynomial vector $\mathbf{s}$ has uniformly random coefficients in the range $[-\eta, \eta]$. The symbol $\perp$ means an invalid value.

TABLE I
PARAMETERS OF DILITHIUM NIST ROUND 3

NIST Security Level		II	III	V
Param.	Meanings	Values		
q	Modulus	8380417	8380417	8380417
d	Power2Round param.	13	13	13
τ	Number of '$\pm$1' in c	39	49	60
γ_1	$\mathbf{y}$ coefficient range	2^{17}	2^{19}	2^{19}
γ_2	Rounding range	95232	261888	261888
(k, ℓ)	Dimensions	(4, 4)	(6, 5)	(8, 7)
η	Secret key range	2	4	2
β	$\tau \cdot \beta$	78	196	160
w	Number of '1' in $\mathbf{h}$	80	55	75

B. The Dilithium Signature

The framework of Dilithium signature [18] is shown in Alg. 1. The algorithm consists of three procedures, i.e. key generation, signature generation, and signature verification. In the Alg. 1, functions like Power2Round, HighBits, LowBits, MakeHint and UseHint are used to reduce the size of signature or key by compressing or decompressing intermediate data. Function H indicates a secure hash function. In line 11, $c \in B_\tau$ means the polynomial c has τ coefficients that are '$\pm$1' and other coefficients are '0'.

The key generation procedure allocates the secret key for signature generation and the public key for verification. The critical secret data in the secret key are $\mathbf{s}_1, \mathbf{s}_2$ and $\mathbf{t}_0$. According to line 4 and 5, $\mathbf{s}_1, \mathbf{s}_2$ and $\mathbf{t}$ are associated in an equation. Therefore, as long as the adversary knows any two of the triples, he can recover the secret key. The signature generation procedure contains a reject evaluation, only the parameters that meet the requirements can be output as the signature. As the secret-related operations marked in red, $\mathbf{s}_1, \mathbf{s}_2$ and $\mathbf{t}$ at least multiply with polynomial c for 1 time respectively in the process of signature generation. The verification procedure distinguishes the signature, it returns '1' if the signature verifies correctly.

The security level of the Dilithium can be adjusted by parameters provided in Tab. I. Security level II is equivalent to SHA-256/SHA3-256 collision search. Security level III is equivalent to AES-192 key search, and security level V is equivalent to AES-256 key search.

C. Polynomial Multiplication

Polynomial vector multiplication is computed polynomial by polynomial. As for the polynomial multiplication such as $c\mathbf{s}_1$, $c\mathbf{s}_2$ and $c\mathbf{t}_0$, there are several common algorithms. Such as the schoolbook algorithm, the efficient sparse vector multiplication algorithm, and the NTT algorithm. Among them, the NTT algorithm is the most widely used polynomial multiplication algorithm in lattice-based cryptography. The parameters of Dilithium conform to the classical NTT, and NTT has been dedicated embedded into the Dilithium algorithm.

NTT-based polynomial multiplication is shown in Alg. 2. Firstly, the two input polynomials c and s are transferred to the NTT domain $\hat{c}$ and $\hat{s}$ by the forward NTT algorithm. Then the two polynomials execute multiplication in the NTT domain point by point thus generating $\hat{t}$. Finally, $\hat{t}$ is transformed from the NTT domain to the ordinary domain with the Inverse NTT (INTT) algorithm.

Algorithm 2: NTT-based polynomial multiplication

Poly_Mul(c, s)
1 $\hat{c} := \mathrm{NTT}(s)$
2 $\hat{s} := \mathrm{NTT}(s)$
3 $\hat{t} = \mathrm{POLY\_PWM}(\hat{c}, \hat{s})$; /* Polynomial point-wise-multiply */
$t := \mathrm{INTT}(\hat{t})$
4 return t

D. The Adversary Model

For a legitimate signature device, the adversary can impersonate a normal user to input messages and request signatures. The generated signature is public to all verifiers.

The adversary aims to forge a digital signature or obtain the secret key, while the latter is more difficult but destructive. Forgery attacks only require partial knowledge of the secret key, and the generated forgery signature may have a higher decryption failure probability than a normal signature. Key recovery attacks can make the adversary obtain the same ability as a legitimate signature device.

In this work, the adversary tries to recover the secret key without invading the device nor profiling the templates in advance. To achieve this goal, CPA is executed. There are five steps for the adversary to carry out a classical CPA.

1) Choose an appropriate intermediate value as the PoI, which is a function of the key and a known variable.
2) Record the power traces. That is, run signing process for n times, and store the m-length power samples captured each time in a matrix $\mathbf{T}_{n\times m}$.
3) Calculate the intermediate value matrix $\mathbf{V}_{n\times k}$ of the key guesses. Then calculate the intermediate value of the PoI according to the known values and the key in the overall guess space.
4) The power model (such as Hamming weight) is used to map $\mathbf{V}_{n\times k}$ to $\mathbf{H}_{n\times k}$, and each term $\mathbf{H}_{i\times j}$ of $\mathbf{H}_{n\times k}$ is the corresponding Hamming weight of $\mathbf{V}_{i\times j}$.
5) Calculate the correlation coefficients of each column in $\mathbf{H}_{n\times k}$ and $\mathbf{T}_{n\times m}$, and record them in $\mathbf{R}_{k\times m}$. Pearson correlation of columns $\mathbf{H}_i$ and $\mathbf{T}_j$ is computed as Eq. 1, in which $\bar{\mathbf{H}}_i$ and $\bar{\mathbf{T}}_j$ are the average of the column.

$$\mathbf{R}_{i,j} = \frac{\sum_{x=1}^{n}\left(\mathbf{H}_{x,i} - \bar{\mathbf{H}}_i\right)\cdot\left(\mathbf{T}_{x,j} - \bar{\mathbf{T}}_j\right)}{\sqrt{\sum_{x=1}^{n}\left(\mathbf{H}_{x,i} - \bar{\mathbf{H}}_i\right)^2 \cdot \sum_{x=1}^{n}\left(\mathbf{T}_{x,j} - \bar{\mathbf{T}}_j\right)^2}} \quad (1)$$

The index corresponding to the maximum value in the matrix $\mathbf{R}$ reveals the time corresponding to the target operation and the key used by the device.

In the above steps, the computational overhead is linearly related to the number of traces n, the number of samples m, and the key guess space k.

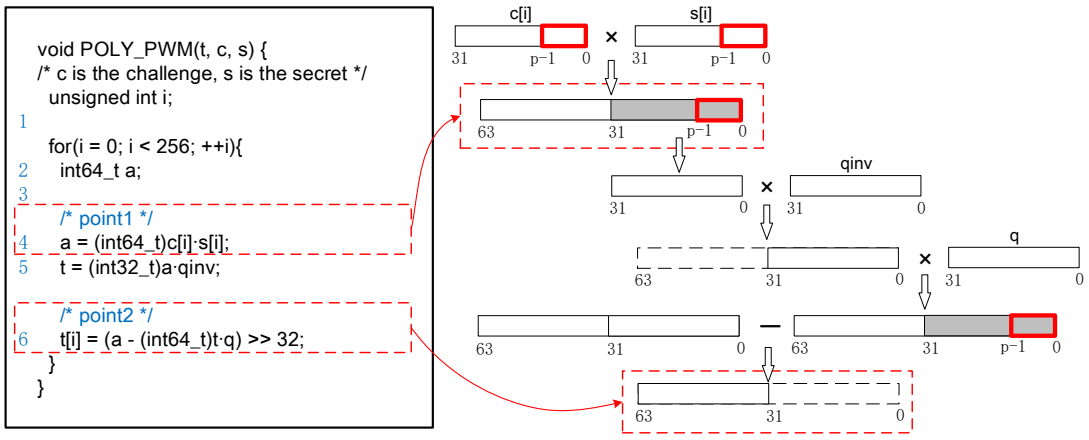


Fig. 1. C code snippet and data flow of polynomial point-wise multiplication in Dilithium Signature

III. The Proposed Side-Channel Attacks

In this section, the vulnerability of NTT-based polynomial multiplication is analyzed. A conservative CPA scheme is introduced, and the parameter algebraic characteristics can be utilized to minimize the key guess space. To further accelerate the attack run-time, a fast two-stage CPA scheme is proposed. Then the adversary can combine the two schemes to juggle the attack's execution time and success rate.

A. The PoI

The point-wise multiplication procedure is appropriate to execute CPA. In this step, the secret key is multiplied by a public challenge c. Even though the coefficients are operated in the NTT domain, the obtained secret polynomial could be easily transformed to normal domain with the INTT algorithm. The polynomial point-wise multiplication algorithm, i.e. POLY_PWM, in the reference implementation is shown on the left side of Fig. 1. Since the coefficients are operated sequentially, the adversary could perform CPA on each coefficient in turn. Line 4-6 in the inner loop is the Montgomery Modular Multiplication (MMM). It provides a low-cost approach to multiply two factors and then modular reduce it to the range $(-q, q)$.

The right side of Fig. 1 shows the data flow of MMM, where $c[i]$ and $s[i]$ are the i-th coefficients of challenge and secret key polynomials. Note that they have been converted to NTT domain in this algorithm. q and $qinv$ are fixed constants, where $qinv = q^{-1} \bmod 2^{32}$. The squares represent 64-bit or 32-bit integers, and the numbers below them indicate the binary width. The data corresponding to the dotted box in the figure is finally discarded. The operation process shown in the figure involves three multiplication operations and one subtraction operation.

In the figure, point1 and point2 are marked by red dotted boxes. The point1 is the straightforward multiplication result of $c[i]$ and $s[i]$. As a common problem, this point may cause false positives. For example, the $s[i]$ is hex(0000FF00), then the key guesses $s[i] >> 1$ = hex(00007F80) is hard to distinguish with the power model because they always show a $bit\text{-}shifting$ relationship and have similar Hamming weight after multiplicating the same $c[i]$. So there are always many key guesses that show correlation peaks. The adversary cannot judge which peak corresponds to the correct key. Similarly, the second and third multiplication results also show false positives. The point2 in Fig. 1 is the modular reduction output, this point is the result of subtraction. Since the $bit\text{-}shifting$ relationship is eliminated, the issue of the false positive in point1 is solved. According to the characteristics of constant q and $qinv$, the gray 32 bits of point1 and multiplication intermediate value in line 6 will be the same. So the least significant 32 bits of point2 should all be '0', and they do not affect the Hamming weight.

```
    POLY_PWM:
1       push   {r4, r5, r6, r7}
2       ldr    r7, .L6
3       subs   r6, r1, #4
4       subs   r0, r0, #4
5       subs   r2, r2, #4
6       add    r1, r1, #1020
    .L2:
7       ldr    r4, [r6, #4]!
8       ldr    r3, [r2, #4]!
9       smull  r4, r5, r4, r3  /*  {r5, r4} = r3 · r4; */
10      rsb    r3, r4, r4, lsl #3
11      add    r3, r4, r3, lsl #10
12      add    r3, r4, r3, lsl #13
13      smlal  r4, r5, r7, r3  /* {r5, r4} = ( {r5, r4} - r3 · r7)  */
14      cmp    r1, r6
15      str    r5, [r0, #4]!   /* Store r5 */
16      bne    .L2
17      pop    {r4, r5, r6, r7}
18      bx     lr
    .L7:
19      .align 2
    .L6:
20      .word  -8380417     /* q = 8380417 */
```

Fig. 2. Register usage in the assembly code snippet of polynomial point-wise multiplication

To analyze the register usage of the POLY_PWM function, the C code is compiled with the arm-none-eabi-gcc compiler

for a 32-bit ARM Cortex-M4 core. The corresponding assembly code is shown in Fig. 2. Lines 9 to 13 in Fig. 2 show the assembly code snippet of Montgomery modular multipliers. The 'smull' instruction performs signed multiplication of $c[i]$ and $s[i]$. The second multiplication is replaced by shifts and additions. 'smlal' instruction performs signed multiplication and accumulation. The final result in the 32-bit register r5 is stored in line 15. The data registering and storage operation in line 13 and line 15 confirm the power leakage of point2.

B. A Conservative Scheme

The point2 is an ideal PoI for CPA so that a straightforward CPA scheme is feasible. Further, the time spent in executing CPA is also an important factor to illustrate the threat. For the Dilithium algorithm, the modulus q is 8380417, which is 23 bits in binary. According to the description in Alg. 2, a secret polynomial is transformed to NTT domain before the point-wise multiplication process. In the reference implementation, the forward NTT algorithm omits the modular reduction in the modular addition and subtraction operation to reduce computation load. After the 8-stage recursion for the 256-dimension polynomial, coefficients in the NTT domain are in the range of $[-\eta - 8(q-1), \eta + 8(q-1)]$. Because the word length is 32 bits which is enough to avoid overflow, only the modular reduction after multiplication is necessary to ensure correctness. However, the omitted modular reduction increases the range of polynomial coefficients. That is to say, the size of the key guessing table is close to 2^{27} in NTT domain. This makes CPA on Dilithium more difficult than that of symmetric cryptography such as the Advanced Encryption Standard (AES), which could be analyzed byte by byte (2^8).

As described in section II-D, there is a linear relationship between operation load and the key guess space. The large key guessing table will make the execution of CPA time-consuming. For the correct key ck, there exists $ck \pm xq \in [-\eta-8(q-1), \eta+8(q-1)]$, with $x \in \mathbb{Z}$. Since the INTT works in $\mathbb{Z}_q$, coefficients with redundant q in the NTT domain maps to the same polynomial in the normal domain. Thus, $ck \pm xq$ would be equivalent to ck for the adversary. Moreover, the modular reduction results of $ck \pm xq$ and ck are the same, so that the values registered in r5 are the same and they would both show peaks on the correlation plot. Thus, the adversary can focus on the range $[0, q-1]$.

The value in the register r5 is a signed 32-bit integer. The complement of a negative number is obtained by inverting all bits in it, i.e., transferring the '0'/'1' bit to '1'/'0' and then adding 1 on the number. Since the Hamming weight of a negative number is approximately opposite to the corresponding positive number, $-ck \pm xq$ also show correlation peaks. However, the peak polarity of the negative numbers would be the opposite. Thus the adversary can distinguish $-ck \pm xq$ according to whether the peak is positive or negative. Therefore, a wise adversary could reduce the key guess space to $[0, q/2]$. Firstly, there must be a correlation peak in this interval, either corresponding to $ck \pm xq$ or $-ck \pm xq$. Secondly, the adversary can infer $ck \pm xq$ from the peak polarity. If the peak polarity is matched, the peak column index key_{peak} is $ck \pm xq$. If the polarity is opposite, the $-key_{peak}$ is $ck \pm xq$. After analyzing all the 256 coefficients in the polynomial, the adversary could recover the normal domain polynomial with INTT. In this way, this conservative scheme can reduce the key guess space to less than 2^{22}, which would accelerate the CPA computation process by 32 times.

C. A Fast Two-Stage Scheme

The proposed conservative scheme can be realized on general computers with a reasonable delay. However, considering that each polynomial has 256 coefficients and each polynomial vector has k or ℓ polynomials, it is still time-consuming to recover the whole key. This part focuses on further reducing the execution time of CPA to make the attack efficient to implement.

Generally speaking, points with inherent false positives are not ideal for CPA. For point1, even if the adversary executes CPA with lots of traces, it is difficult to eliminate the false positive key guesses. Inspired by [19], CPA on large number multiplication can also be attacked bit-slice-wise separately. As shown on the right side of Fig. 1, the Least Significant p-Bit (LSB-p) of several intermediate values are marked as red squares. In the process of calculating the multiplication of point1, the LSB-p bits of point1 is only associated with the LSB-p bits of both $c[i]$ and $s[i]$. Because the key guess space of LSB-p of $s[i]$ (denoted by $s[i]_{\mathrm{LSB}p}$) is significantly smaller than the whole key space, the CPA process on point1$_{\mathrm{LSB}p}$ could be fast. The adversary can combine the two PoIs to accelerate attack run-time. The fast two-stage CPA scheme is proposed as follows.

- **Stage1**. In this stage, the adversary executes CPA by targeting the LSB-p of point1. This stage aims to eliminate wrong key guesses by selecting several LSB-p candidates with high correlations. Different from the ideal CPA, due to the false positives and the noise from other high-order bits of point1, there are more than one peaks appears and the highest peak may not be the correct $s[i]_{\mathrm{LSB}p}$. Even though the adversary cannot recognize the false positives in this stage, he can record these candidates since the correct LSB-p quite probably ranks in the top ones. If the correct LSB-p is included in the range of LSB-p candidates, stage 1 $hits$. The adversary needs a high hit rate to let this stage make sense.
- **Stage2**. In this stage, the adversary first constructs a key guess list using the candidates of $s[i]_{\mathrm{LSB}p}$ obtained in stage 1. Then he can execute the CPA by targeting point2 to recover the complete secret coefficient. If stage 1 $hits$, the correlation plot shows an obvious peak, otherwise there is no peak. Since stage 1 narrows the key guess table, CPA in this stage costs a relatively low computation load.

On the computational overhead, variable p is a critical parameter to optimize algorithm complexity. For a fixed number of sampling points m, the computational load of stage 1 is related to $(n_{stage1} \cdot 2^p) \cdot m$. In the stage 2, the load is related

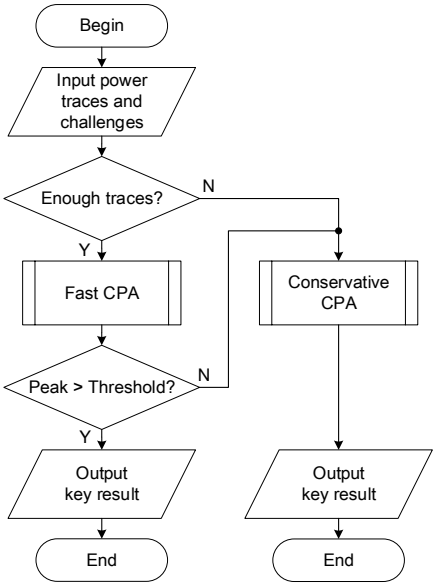


Fig. 3. Flow chart of the hybrid scheme

to $(n_{cdd} \cdot n_{stage2} \cdot 2^{(27-p)}) \cdot m$, in which n_{cdd} is the number of candidates in stage 1. The overall goal is to keep a high hit rate and minimize the sum of the overhead of stage 1 and stage 2. This can be achieved by tuning parameters like n_{stage1}, n_{stage2} and p. Related analysis and the test result will be described in Section IV.

D. A Hybrid Scheme

The Fast two-stage CPA scheme can reduce the computation delay, but it is a probabilistic approach to recover the coefficient. It may fail when stage 1 does not hit. Therefore, if the adversary uses this scheme standalone, he bears the loss of success rate. In practice, the adversaries can combine the conservative and fast schemes to optimize the run-time and also maintain a high success rate.

As shown in Fig. 3, the adversary first collects a series of power traces of polynomial point-wise multiplication operation. Since stage 1 of the fast scheme needs a relatively large number of traces to ensure a high hit rate, the adversary should judge whether it is feasible to execute the fast scheme. If the number of power traces is insufficient, he can directly execute the conservative CPA. Otherwise, the adversary can use the fast two-stage scheme and further judge whether it hits by comparing the peak value with a threshold. Since there is no false positive for point2, the correct $s[i]_{\text{LSB}p}$ is missed if there is no obvious peak in stage 2. If so, the adversary can analyze this coefficient again with the conservative scheme.

IV. Experiment Results

This section constructs the experimental environment and analyzes the effect of the attack. The results verify that the conservative scheme can achieve 99.99% confidence with 157 power traces, and the hybrid scheme can further improve the efficiency of CPA.

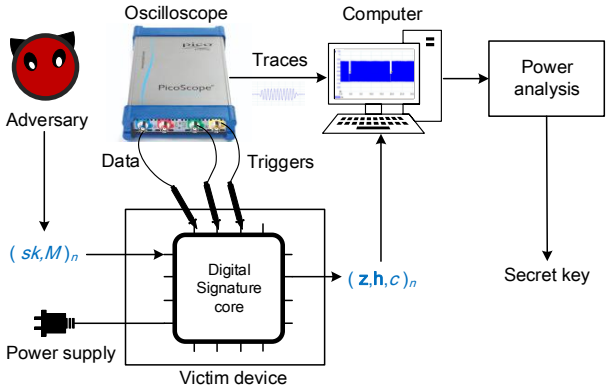


Fig. 4. Schematic diagram of the CPA experimental environment

A. Experimental Setup

The experiment imitates the adversary's ability as shown in Fig. 4. The victim device, CW308T, is powered by an ordinary power supply. The board is equipped with an STM32F405 chip that adopts ARM Cortex-M4 core. The chip runs the round 3 reference Dilithium code on 168MHz. The arm-none-eabi-gcc compiles the target programs with -O3 level optimization. A PicoScope 6402C oscilloscope is deployed to capture the power traces. It captures 1.25G samples per second with 250MHz bandwidth. Three probes connect to the pins of the victim device, one of them is used to capture the sampled data, and the other two are used as triggers. The captured data is transferred to a computer with a USB cable. The CPA process is done on a computer with an Intel i7-10750H processor and 64GB DDR4 memory. The CPA programs are implemented with MATLAB R2016a.

The Dilithium signature includes the abort process in lines 14 and 19 of Alg. 1. This means if the parameters do not meet requirements, the signature will be discarded. The signature generation process keeps trying, and only the signature $(\mathbf{z}, \mathbf{h}, c)$ that meets the requirements will be output. Therefore, to capture trace corresponding to the final signature, trigger A is synchronized with the completion of signature generation, and trigger B is synchronized with the target operation.

Fig. 5 presents a single trace of the Dilithium signature. The adversary first locates the end of the trace according to trigger B in the top sub-graph, and then the high level of the trigger A next to it corresponds to the last loop, as the middle sub-graph in Fig. 5 shows. Finally, for the 256-dimensional point-wise multiplication, the waveform can be distinguished to obtain 256 cycles of MMM. The bottom sub-graph in Fig. 5 presents a piece of the waveform. With these traces, the adversary can analyze each coefficient $s[i]$ in turn. Then the entire polynomial can be obtained. Similarly, the secret key polynomial vectors $\mathbf{s}_1, \mathbf{s}_2$ and $\mathbf{t}_0$ can be recovered.

B. Results of the Conservative Scheme

We first execute the conservative CPA scheme on the secret polynomial coefficient. The reference [20] uses the parameter $\rho(ck, ct)$ to estimate how many traces the adversary needs

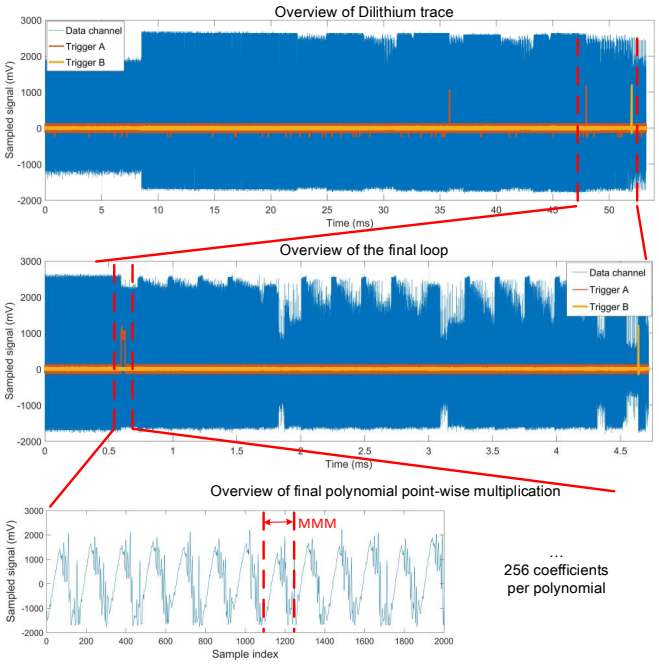
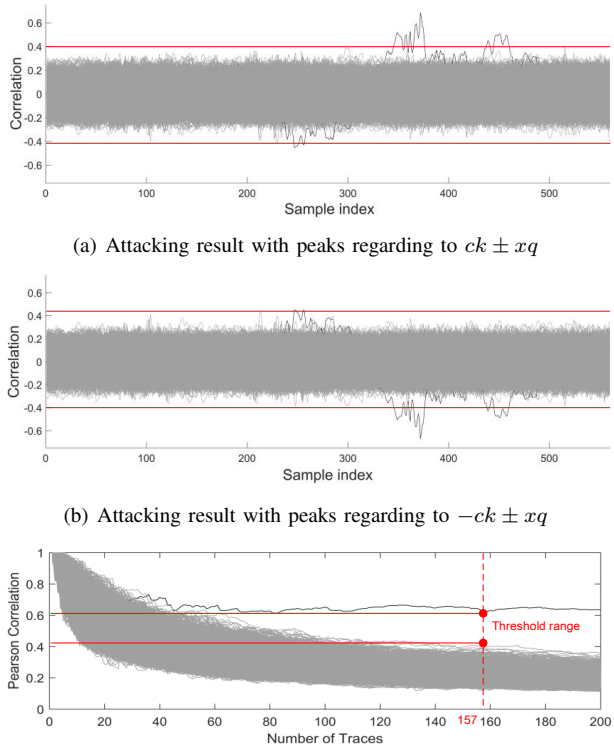


Fig. 5. Power traces of the reference implementation of Dilithium on STM32F405



(a) Attacking result with peaks regarding to $ck \pm xq$

(b) Attacking result with peaks regarding to $-ck \pm xq$

(c) Evaluating number of traces requirement

Fig. 6. Result of the conservative scheme.

to recover the secret key. The parameter $\rho(ck, ct)$ means the Pearson correlation of the power trace and power model with the correct key (ck) at the correct time (ct), while the Pearson correlation coefficient of other key guesses should converge to 0. We estimate the $\rho(ck, ct)$ with 100k traces, and find $\rho(ck, ct)$ can reach at least 0.4. According to the hypothesis testing principle, 157 power traces are sufficient to recover the correct key with 99.99% confidence [20].

For our experimental device, the maximum correlation peak value of the correct key is always positive. As shown in Fig. 6(a), the correct key shows obvious peaks and the largest absolute value of the peak is positive. Because of the complement property of negative numbers, the Hamming weight of $-ck \pm xq$ is opposite to that of $ck \pm xq$. Therefore, the correlation coefficient curve of $-ck \pm xq$ will also have obvious peaks, but the maximum value of the peaks is negative, as shown in Fig. 6(b). When the adversary reduces the range of key guessing to [0, q/2], He can find the correlation coefficient with the largest absolute value, and then recover the coefficient according to the positive or negative of the peak value. If the peak value is positive, then the correct key is the key guess corresponding to the peak, otherwise, the correct key is the opposite number of the key guess modulus q corresponding to the peak. As shown in Fig. 6(c), the correct key and the wrong key guess can be clearly distinguished with 157 traces. The correlation of the correct key coefficient is over 0.60, and the correlation of other key guesses is below 0.45.

C. Results of the Hybrid Scheme

As explained in Section III-D, if the adversary can capture enough traces, the hybrid scheme gives high priority to the fast scheme. Thus the run-time acceleration makes sense. The benefits of our hybrid scheme mainly depend on the following two aspects. The first is the hit rate of fast scheme. The second is the run-time acceleration of the fast scheme. According to the calculation process of fast scheme in III-C, we estimate a simplified computational complexity model as the formula $(n_{stage1} \cdot 2^p + n_{cdd} \cdot n_{stage2} \cdot 2^{(27-p)}) \cdot m$, where n_{stage1}, n_{cdd}, p and $(27 - p)$ are positive integers. Parameter n_{stage2} can be fixed as 157, m is also a constant. In the formula, n_{stage1} and n_{cdd} have positive influence on hit rate, while p has a negative influence. On the other hand, increasing n_{stage1} and n_{cdd} have negative effects on speed. By the way, n_{stage1} also depends on the adversary's capability to collect traces. For a specific total number of traces n_{stage1}, appropriate n_{cdd} and p can be selected to minimize the load.

As shown in Tab. II, we test a series of n_{stage1}, n_{cdd} and p to balance the computational load and hit rate. For each parameter setting, we analyze 256 coefficients to estimate the hit rate and overall acceleration with the hybrid scheme. As for the threshold setting, according to Fig. 6, the threshold can be set as 0.47 as an example. We first try to find a appropriate n_{stage1} with a constrained time for the fast scheme phase. Among the test cases, the hybrid a, b, c, e, g, h are set as similar time cost on the fast scheme phase in hybrid scheme. As a result, the parameter set $\{n_{stage1}, n_{cdd}, p\} = \{10000, 12, 12\}$ shows the best acceleration effect. We further tune the n_{cdd} with fixed 10000 traces as the test cases hybrid d and f. As we analyzed, the hit rate improves with more

TABLE II
EFFICIENCY COMPARISON OF KEY RECOVERY ATTACKS WITH DIFFERENT PARAMETER SETS

Scheme	No. Traces	n_{cdd}	p	Hit Rate[1]	Acceleration
Conservative	157	NA	NA	NA	1.0
Hybrid a	1500	80	15	77.73%	3.19
Hybrid b	3000	40	14	93.75%	6.56
Hybrid c	5000	24	13	92.97%	6.34
Hybrid d		8		91.02%	6.63
Hybrid e	10000	12	12	96.10%	**7.77**
Hybrid f		16		**96.48%**	6.83
Hybrid g	15000	8	12	94.53%	7.20
Hybrid h	20000	6	11	94.53%	7.06

[1] The hit rate is the probability that the key can be recovered successfully by using only fast scheme.

candidates. However, the acceleration rate decreases because the time consumption of the fast scheme phase itself increases. All in all, our best parameter set achieves 7.77× acceleration compared with the conservative scheme standalone as the first entry. As a baseline, the conservative scheme takes 6572 seconds to break one coefficient. As for breaking the secret polynomial vector $\mathbf{s}_1, \mathbf{s}_2$ for Dilithium-II, our hybrid scheme can save 3403 hours compared with the baseline. Note that the efficiency of our scheme could be better if it adopts a more precise parameter adjustment, but this is not the focus of this work.

We also try to use the result of lines 10, 11, and 12 in Alg. 2 as point1, but the peaks are less obvious, it probably due to the successive add instructions execute in a compact pipeline which increases the noise.

V. CONCLUSION

In this work, a conservative scheme and a fast two-stage attack strategy are proposed. Adversaries can combine these two schemes to achieve an efficient hybrid CPA attack. Experiments show that it can achieve 99.99% confidence with a reasonable amount of power traces. Benefit from our strategy, the adversary can recover the secret key with 7.77× acceleration. This work points out that unprotected NTT-based polynomial multiplication is fragile. The idea proposed in this paper can be easily applied to other NTT-based cryptography implementations, such as Kyber and NewHope. Since they have a compact range of coefficients, the computational overhead for the CPA of PKE/KEM is lower.

This work only analyzes the side-channel leakage of the reference implementation. As common measurements, masking and shuffling countermeasures are still effective to avoid this attack, although these techniques cost additional time and resources. We will also continue to improve our CPA technique and try to analyze the side-channel resistance implementations.

ACKNOWLEDGMENT

This work was partially supported by National Key R&D Program of China (Grant No.2020YFB1806205), National Natural Science Foundation of China (No. 61872357 and No. 61802396), and Research Program of BJCA (BJCA2020-YF-0300). Emre Karabulut's and Aydin Aysu's contributions are not supported by any agencies or companies.

REFERENCES

[1] R. L. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[2] C. Schnorr, "Efficient identification and signatures for smart cards," in *Proc. CRYPTO 1989, Santa Barbara, USA, Aug. 1989*, pp. 239–252.

[3] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *35th Annual Symp. on Foundations of Computer Science, Santa Fe, New Mexico, USA, Nov. 1994*, pp. 124–134.

[4] C. Schnorr and M. Euchner, "Lattice basis reduction: Improved practical algorithms and solving subset sum problems," *Math. Program.*, vol. 66, pp. 181–199, 1994.

[5] R. Cramer, L. Ducas, and B. Wesolowski, "Short stickelberger class relations and application to ideal-svp," in *Proc. EUROCRYPT 2017, Paris, France, Apr. 2017*, pp. 324–348.

[6] P. Kirchner and P. Fouque, "Revisiting lattice attacks on overstretched NTRU parameters," in *Proc. EUROCRYPT 2017, Paris, France, Apr. 2017*, pp. 3–26.

[7] V. Migliore, B. Gérard, M. Tibouchi, and P. Fouque, "Masking dilithium - efficient implementation and side-channel evaluation," in *Proc. ACNS 2019, Bogota, Colombia, Jun. 2019*, pp. 344–362.

[8] I. Kim, T. Lee, J. Han, B. Sim, and D. Han, "Novel single-trace ML profiling attacks on NIST 3 round candidate dilithium," *IACR Cryptol. ePrint Arch.*, vol. 2020, p. 1383, 2020.

[9] R. Primas, P. Pessl, and S. Mangard, "Single-trace side-channel attacks on masked lattice-based encryption," in *Proc. Cryptographic Hardware and Embedded Systems CHES 2017, Taipei, Taiwan, Sep. 2017*, pp. 513–533.

[10] P. Pessl and R. Primas, "More practical single-trace attacks on the number theoretic transform," in *Proc. LATINCRYPT 2019, America, Santiago de Chile, Chile, Oct. 2019*, pp. 130–149.

[11] P. C. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Proc. CRYPTO 1999, Santa Barbara, USA, Aug. 1999*, pp. 388–397.

[12] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *Proc. Cryptographic Hardware and Embedded Systems CHES 2004, Cambridge, MA, USA, Aug. 2004.*, pp. 16–29.

[13] A. P. Fournaris, C. Dimopoulos, and O. G. Koufopavlou, "Profiling dilithium digital signature traces for correlation differential side channel attacks," in *Proc. Embedded Computer Systems: Architectures, Modeling, and Simulation SAMOS 2020, Greece, Jul. 2020*, pp. 281–294.

[14] Y. Liu, Y. Zhou, S. Sun, T. Wang, R. Zhang, and J. Ming, "On the security of lattice-based fiat-shamir signatures in the presence of randomness leakage," *IEEE Trans. Inf. Forensics Secur.*, vol. 16, pp. 1868–1879, 2021.

[15] A. Aysu, Y. Tobah, M. Tiwari, A. Gerstlauer, and M. Orshansky, "Horizontal side-channel vulnerabilities of post-quantum key exchange protocols," in *2018 IEEE International Symp. on Hardware Oriented Security and Trust, HOST 2018, Washington, DC, USA, Apr. 2018*, pp. 81–88.

[16] P. Ravi, M. P. Jhanwar, J. Howe, A. Chattopadhyay, and S. Bhasin, "Side-channel assisted existential forgery attack on dilithium - A NIST PQC candidate," *IACR Cryptol. ePrint Arch.*, vol. 2018, p. 821, 2018. [Online]. Available: https://eprint.iacr.org/2018/821

[17] F. Aydin, P. Kashyap, S. Potluri, P. Franzon, and A. Aysu, "Deepar-sca: Breaking parallel architectures of lattice cryptography via learning based side-channel attacks," in *Proc. Embedded Computer Systems: Architectures, Modeling, and Simulation SAMOS 2020, Greece, Jul. 2020*, pp. 262–280.

[18] S. Bai, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehléy. (2021, Feb.) Crystals-dilithium algorithm specifications and supporting documentation (version 3.1). [Online]. Available: https://pq-crystals.org/dilithium/data/dilithium-specification-round3-20210208.pdf

[19] M. Tunstall, N. Hanley, R. McEvoy, C. Whelan, C. Murphy, and W. Marnane, "Correlation power analysis of large word sizes," in *IET Irish Signals and Systems Conf. (ISSC)*, 2007, pp. 145–150.

[20] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Berlin, Heidelberg: Springer-Verlag, 2007.