SHIFT SNARE: Uncovering Secret Keys in FALCON via Single-Trace Analysis

***Abstract*—This paper presents a novel *single-trace* side-channel attack on FALCON—a lattice-based post-quantum digital signature protocol recently approved for standardization by NIST. We target the discrete Gaussian sampling operation within the FALCON key generation scheme and use a single power measurement trace to succeed. Notably, negating the ‘shift right 63-bit’ operation (for 64-bit values) leaks critical information about the ‘-1’ vs. ‘0’ assignments to intermediate coefficients. These leaks enable full recovery of the generated secret keys. The proposed attack is implemented on an ARM Cortex-M4 microcontroller running both reference and optimized software implementation from FALCON’s NIST Round 3 package. Statistical analysis with 500k tests reveals a per coefficient success rate of 99.9999999478% and a full key recovery success rate of 99.99994654% for FALCON-512. This work highlights the vulnerability of current software solutions to single-trace attacks and underscores the urgent need to develop single-trace resilient software for embedded systems.**

***Index Terms*—side-channel attacks, post-quantum cryptography, NTRU, FALCON**

I. Introduction

Widely adopted encryption schemes such as RSA [1] and elliptic curve cryptosystems [2] rely on problems such as the discrete logarithm [3] and integer factorization [4]. Unfortunately, quantum computers have been proven to solve these problems with exponential speedup [5], which motivates the need for alternatives.

To address this issue, the National Institute of Standards and Technology (NIST) initiated a standardization process for quantum-resilient (also known as post-quantum) cryptographic schemes that can withstand quantum cryptanalysis [6]. This standardization process selected three digital signature schemes: CRYSTALS-Dilithium [7], SPHINCS+ [8], and FALCON [9]. NIST selected FALCON due to its small signature sizes, which make it especially favorable for embedded systems applications.

While the chosen algorithms are claimed to be mathematically robust, their practical implementations may be vulnerable to side-channel attacks. These attacks exploit implementation characteristics such as execution time, power consumption, and electromagnetic radiation to extract secret values [10]. An attacker can execute these attacks with only a few side-channel measurements from the physical device [11], [12], [13], [14]. The most extreme form of these attacks, known as *single-trace attacks* (a.k.a., simple power analysis), allows the adversary to extract secrets using measurements from just a single program execution. These attacks are perilous because single-trace measurements bypass common defenses like masking [12]. Moreover, they can also target subroutines such as key generation that generate a new secret each time it executes. FALCON is especially suitable for embedded system deployment, making it a prime target for side-channel attacks. Given the imminent real-world deployment of NIST’s post-quantum algorithms, there is a critical need to expose these attacks and inform effective defenses.

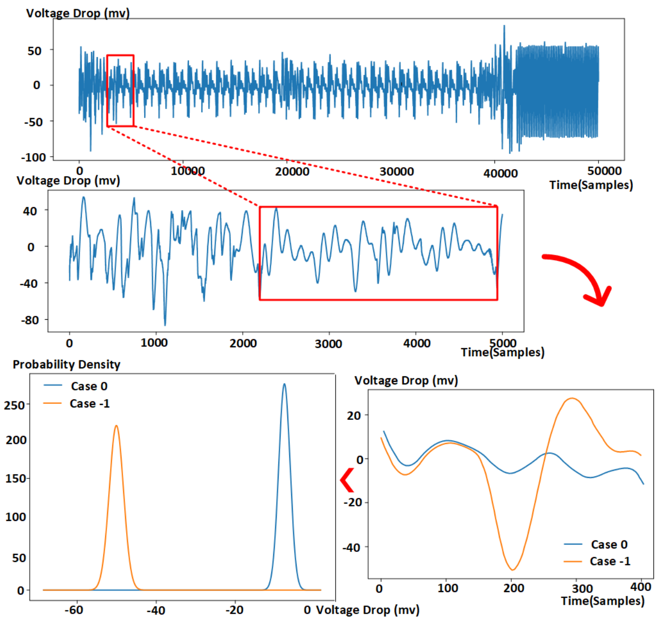

Fig. 1. Visual demonstration of the vulnerability: The top figure illustrates the power consumption profile of the device during the key generation process. The middle figure provides a zoomed-in view of the power consumption during a specific vulnerable segment in the code. The bottom right figure offers a further magnified view, along with the average power consumption for two distinct cases of secret value assignment (0 versus -1). The bottom left figure presents the results of a univariate Gaussian model of these two classes over 500k trials. The analysis reveals that different assignments of secret intermediates cause significant variations in power consumption, highlighting the vulnerability.

Prior works on the single-trace side-channel analysis of lattice cryptosystems have targeted several vulnerable components of the algorithm, such as the number theoretic transform (NTT) [15], [16], [17], discrete Gaussian sampling [18], polynomial multiplication [19], [20], [21], message encoding/decoding [22], [23], [24], [25], and other elements such as ω-small sampling [26], cumulative distribution table (CDT) sampling [27], [28], Fujisaki-Okamoto Transform [29]. Although FALCON incorporates some of these components, it also includes distinct operations such as fast Fourier sampling and floating-point arithmetic. Earlier attacks cannot be trivially extended to the implementation of these units. Side-channel vulnerabilities in FALCON's software implementation remain largely unexplored. Previous studies have investigated multi-trace attacks on FALCON [30], [31], [32]; however, its susceptibility to single-trace attacks remains largely unknown, aside from a vulnerability [33] on base sampling. This gap necessitates further investigation.

In this paper, we reveal a *new* single-trace side-channel vulnerability that enables full recovery of the session's secret key using only side-channel information. This vulnerability resides in the discrete Gaussian sampling subroutine of FALCON's reference software implementation and is orthogonal to the vulnerability disclosed in the prior attack [33]. Compared to previous work, our attack eliminates the need for lattice reduction and avoids computationally intensive post-processing.

We first present the leaky operations used in the implementation and show how they leak important intermediate '0' and '-1' value assignments. Furthermore, we discuss the algorithm and demonstrate how these assignments can lead to full secret key recovery. We demonstrate how to locate these vulnerable operations in the power trace. We use statistical methods to extract these assignments with high accuracy and efficiency. Finally, we provide visualizations of our attack results and quantify our attack's success rate.

Figure 1 illustrates the vulnerability and its significance. The top graph presents a power measurement trace recorded during the discrete Gaussian sampling process. In the second graph from the top, we highlight the areas of interest where the leakage occurs. The bottom right graph offers a further magnified view and shows the two distinct cases, '0' and '-1', corresponding to secret value assignments. The bottom left graph displays the results of a univariate Gaussian model constructed on the two classes using the maximum leakage point, revealing a separation between the different secret value assignments. This clear separation shows the vulnerability of FALCON.

The contributions of this paper are as follows:

- We reveal a new side-channel vulnerability in FALCON's key generation process that could lead to full secret key recovery using only a single power measurement.
- We demonstrate the identification of leakage points and derive models to assess the success rate.
- We applied the attack on an off-the-shelf device containing an ARM Cortex-M4 microcontroller running reference and optimized software implementations from the FALCON NIST submission package. Our attacks extracted the targeted coefficient with a success rate of 99.9999999478% per secret variable and a full key recovery success rate higher than 99.99989309%. The attack remains effective in both the reference and optimized implementations, as the same exploitable leakage exists in both versions.

II. BACKGROUND

This section provides an overview of FALCON and explains its secret polynomials. For brevity, we omit the details in FALCON's key generation process. Following this, we outline our threat model.

A. The Generation of FALCON's Secret Polynomials

FALCON stands for Fast-Fourier Lattice-Based Compact Signature Over NTRU. It is a digital signature scheme designed for the post-quantum era, meaning it would take a quantum computer a significant amount of time to break the mathematical trapdoors used in this algorithm. FALCON consists of three main parts: key generation, signature generation, and signature verification. The key generation process defines the NTRU-Lattice components f and g. The session's public key, secret key, and signature are derived from f and g without involving any randomness, as specified in Algorithm 1. This paper focuses on the discrete Gaussian sampling subroutine that generates f and g with the format shown below:

$$f(x) = f_0 + f_1x + f_2x^2 + f_3x^3 + \cdots + f_{n-1}x^{n-1} \qquad (1)$$

$$g(x) = g_0 + g_1x + g_2x^2 + g_3x^3 + \cdots + g_{n-1}x^{n-1} \qquad (2)$$

The coefficients of f and g (namely, $f_0, f_1, ..., f_{n-1}$ and $g_0, g_1, ..., g_{n-1}$) are sampled individually using a discrete Gaussian distribution. The mean of this distribution is 0, while the standard deviation is determined by the degree n and the parameter q. Degree n is defined by the user, which is either 512 or 1024, and q is set to 12289.

The Gaussian sampling method generates values through iterative sampling performed during lines 3 and 4 of Algorithm 1. Within the sampling process, each iteration generates a random number to determine if the sample should be zero or non-zero. If the sample is non-zero, another random value is used to select a threshold-based index from a pre-computed table that aligns with the desired distribution. The sample's sign is then randomly set to positive or negative as the sample mean is 0. The signed sample is accumulated to a final sum, which is returned as the output after multiple iterations. While the mathematical details of this process are beyond the scope of the discussion, the next section will demonstrate how the software implementation facilitates full key recovery.

Algorithm 1 NTRUGen(ϕ, q)

```
Require: A monic polynomial $\phi \in \mathbb{Z}[x]$ of degree $n$, a modulus $q$
Ensure: Polynomials $f$, $g$, $F$, $G$
 1: $\sigma_{f,g} \leftarrow 1.17\sqrt{q/2n}$
 2: for $i \leftarrow 0$ to $n-1$ do
 3:     $f_i \leftarrow D_{\mathbb{Z},\sigma_{f,g},0}$
 4:     $g_i \leftarrow D_{\mathbb{Z},\sigma_{f,g},0}$
 5: end for
 6: $f \leftarrow \sum_i f_i x^i$                    ▷ $f \in \mathbb{Z}[x]/(\phi)$
 7: $g \leftarrow \sum_i g_i x^i$                    ▷ $g \in \mathbb{Z}[x]/(\phi)$
    ...
 8: $(F, G) \leftarrow NTRUSolve_{n,q}(f, g)$   ▷ Compute $F$, $G$ such that $fG - gF = q \mod \phi$
 9: if $(F, G) = \perp$ then
10:     restart
11: end if
        return $f, g, F, G$
```

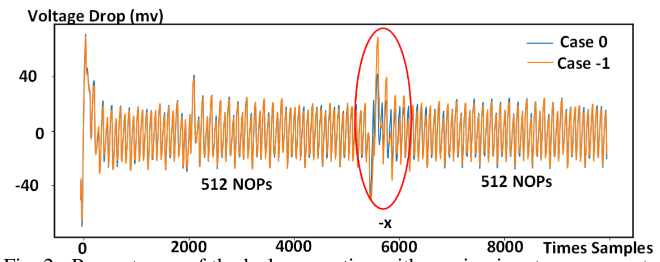

Fig. 2. Power traces of the leaky operation with varying inputs are presented. The trace in blue depicts the power consumption during the leaky operation when the secret is assigned the value '0', while the trace in orange shows the power consumption when the secret is assigned '-1.' A distinct difference in power consumption between the two cases is observable.

B. Threat Model

We adopt the well-established threat model for single-trace side-channel attacks [18], [26], assuming an adversary with physical access to the target device who can measure power consumption during cryptographic operations. The adversary is assumed to possess knowledge of the executing software, approximate the timing of specific computations, and intercept communication channels to capture exchanged public messages. During the characterization phase, the attacker can supply random inputs and analyze the software's power behavior with known values. However, at runtime, they are restricted to capturing a single power trace in their attempt to deduce the entire secret key. Other attacks such as fault injection, buffer overflow, and attacks targeting operating systems are considered out of scope.

III. Underlying Mechanism of the Attack

In this section, we first demonstrate how the target bit shift operation leaks intermediate secret variables. We also present proof-of-concept studies that validate this vulnerability. Subsequently, we explain how these leaked variables enable full recovery of FALCON's secret polynomial.

A. The Operations That Leak

This section describes how negating the 'shift 63-bit' operation can leak the value assignments of 0 and -1 and preliminary results are presented to substantiate this claim.

For a 64-bit variable, the 'right shift 63-bit' operation (expressed as '$(x \gg 63)$' in software) shifts the most significant bit (MSB) to the least significant bit (LSB) and clears all other bits. This operation produces only two possible outcomes: '0' or '1.' The negation of this result changes the value to either '0' or '-1,' represented in two's complement as all 0's or all 1's, respectively. When the processor writes the result, the '-1' case exhibits a Hamming Weight (HW) of 64, while the '0' case has a HW of 0. **Consequently, the '-1' case results in higher power consumption compared to the '0' case.**

Figure 2 illustrates the results of an experiment validating the preceding argument. The experiment involves performing a negation operation on a 64-bit value using the assembly code shown in Listing 1, aligning with the FALCON implementation. The `sbc.w` instruction is employed for two's complement negation (sign inversion) of the 64-bit value. The power consumption of the operation $-(x)$ was measured, with x taking a value of either 1 or 0 (the result of $x \gg 63$).

Listing 1. Assembly instructions corresponding to $-(x)$

```
negs    r2, r2;
sbc.w   r3, r3, r3, lsl #1;
strd    r2, r3, [r7, #24];
```

Listing 2. Gaussian sampling implementation from NIST submission package

```
1   mkgauss(RNG_CONTEXT *rng, unsigned logn){
2       ...
3       for (u = 0; u < g; u ++) {
4           ...
5           r = get_rng_u64(rng);
6           neg = (uint32_t)(r >> 63);
7           r &= ~((uint64_t)1 << 63);
8           f = (uint32_t)((r -
                gauss_1024_12289[0]) >> 63);
9           v = 0;
10          r = get_rng_u64(rng);
11          r &= ~((uint64_t)1 << 63);
12          for (k = 1; k < (sizeof
                gauss_1024_12289)
13              / (sizeof gauss_1024_12289[0])
                    ; k ++){
14              uint32_t t;
15              t = (uint32_t)((r -
                    gauss_1024_12289[k]) >> 63)
                     ^ 1;
16              v |= k & -(t & (f ^1));
17              f |= t;}
18          v = (v ^-neg) + neg;
19          val += *(int32_t *)&v;}
20      return val;}
```

Assembly `NOP` instructions were inserted around the operation to isolate it. We plotted the two graphs on top of each other to show the difference, the blue graph shows the results when x is 0, indicating a peak voltage drop of only 40 mV. The orange graph illustrates the results when x is 1, revealing a peak voltage drop of 70 mV. Additionally, the power spike is more pronounced in the '-1' case compared to the '0' case, demonstrating the vulnerability.

B. Only a Few Variables Needed

This subsection explains how an adversary can recover FALCON's base component f and g using only a few intermediate variables. Recall, that the variables of f and g are generated using a discrete Gaussian sampling process. The reference C code implementation of this process from the NIST submission package is outlined in Listing 2. This subroutine is called n times to generate n variables for the polynomial that forms the base component of the NTRU lattice. The parameter n is the degree of the polynomial specified by the user. This code implementation is composed of a set of nested loops. The outer loop executes twice, while the inner loop executes 26 times for each outer loop execution.

Within Listing 2, line 20 contains the generated variable. To extract the generated secret coefficient, we trace the changes to this variable backward in this subroutine. Line 19 performs a pass-by-value operation followed by a pass-by-reference operation, but numerically it simplifies to `val = val + v`. Consequently, the adversary requires the value of v to deduce the returned result. In Line 18, `v` is XORed with `-neg` and

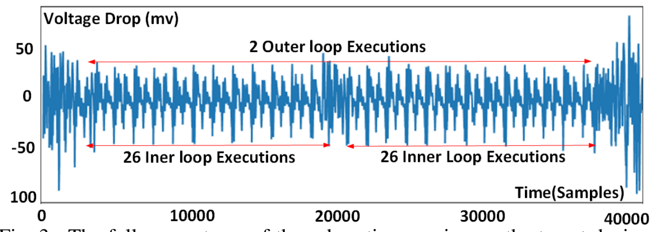

Fig. 3. The full power trace of the subroutine running on the target device is shown. The two outer loop executions and the 26 inner loop executions are clearly observable.

added to `neg`, implying that the adversary must know both `v` and `neg` to determine `val`. We therefore identify the two attack points in this algorithm that will lead to full recovery on f and g. They are line 16 to learn the value of `v` and line 18 to learn the value of `neg` (both highlighted).

Line 16 is our first attack point because the value of `-(t & (f ^ 1))` can only take on '0' or '-1'. The reason is that `t` and `f` (not to be confused with the base lattice component f) can only take '0' or '1' due to the 'shift 63-bit' operation on line 8 and line 15, which clears everything other than the most significant bit (MSB). The negation of `t & (f ^ 1)` thus turns into a negation of '0' or '1'. The negation result, expressed in two's complement, will be all 0s (for 0) or all 1s (for -1). **This will cause a significant power consumption difference of the target device because the Hamming Weight (HW) of these two results differs by 64.** Additionally, in line 16, k is the sequence of the inner loop execution (a known number between 1 and 26). An adversary can infer the value of `v` by exploiting this vulnerability.

Line 18 is our second attack point because `-neg` can only take '0' or '-1'. This is due to the 'shift 63-bit' operation on line 6. The negation of `neg` on line 18 creates a similar vulnerability compared to our first attack point because of the HW difference. An adversary can infer the value of `neg` by attacking this vulnerability.

Since the attack points reside within a loop, and the current loop execution relies on the value of `v` generated from the previous loop iterations. Therefore, the attack success rate must be sufficiently high to ensure successful recovery. An incorrectly inferred intermediate value will result in outcomes that differ from the ground truth.

IV. Exploiting the Found Vulnerability

This section presents the proposed attack strategy for recovering the secret polynomials in FALCON. We will inspect the power trace and discuss how we identify the point of interest. As noted in the previous section, the attack requires a profiling stage in which the adversary has physical access to the hardware and software. During this stage, multiple measurements from different software inputs are used to build a profile that helps determine *when* the leakage occurs. However, once the profile is obtained, the adversary can infer the secret polynomial using a single power measurement.

A. Inspecting the Power Trace

Figure 3 shows the full power trace obtained when executing the discrete Gaussian sampling subroutine. The two outer loop executions and the 26 inner loop executions within each outer loop are distinguishable, reflecting the code's structure shown in Listing 2. We observe a recurring pattern for each inner loop execution approximately every 700 samples, and for each outer loop execution, that is 19000 samples. Since the discrete Gaussian sampling subroutine is executed n times during the key generation process, the resulting power trace patterns will be easily distinguishable.

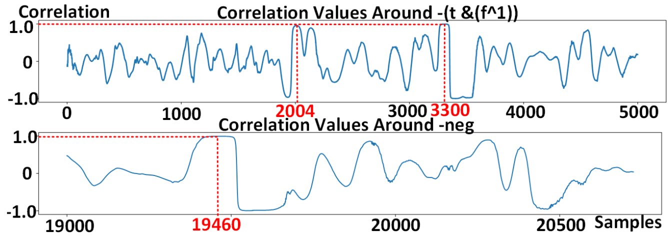

Fig. 4. Correlation power analysis (CPA) results. For the first attack point, the leaky operation occurs at timestamps 2004 and 3300, corresponding to correlation values of 0.996 and 0.977. For the second point, the leaky operation occurs at timestamps 19460, corresponding to a correlation value of 0.992.

B. Pinpointing the Point of Interest

We apply correlation power analysis (CPA) to identify the point of interest (POI). The Pearson correlation values are obtained between the power measurements and a predetermined value set over time. First, we assign values to variable `r` and adjust the values in the matrix `gauss_1024_12289` so that the value of `-(t & (f ^ 1))` in the first and third inner loop executions can be manually controlled. We take 500k measurements, setting `-(t & (f ^ 1))` to '-1' in the first loop and '0' in the third for the first 250k, and reversing these settings for the next 250k. The predetermined value set of 500k numbers has 64s for the first 250k and 0s for the rest, corresponding to the respective power measurements.

When `-(t & (f ^ 1))` is assigned the value -1, it will have a Hamming Weight (HW) of 64. When `-(t & (f ^ 1))` is assigned the value 0, it will have a HW of 0. Due to the power characteristics specified in Section III, this HW distribution will be reflected in the device's power consumption. This leads to a strong correlation between the device's power consumption and the predetermined value set at the time stamp when `-(t & (f ^ 1))` occurs in the first and third inner loop executions. **This means that the Pearson correlation value will be very high at these two timestamps.**

Similarly, for the second leaky operation, we controlled the value assignment of `-neg` and applied Pearson correlation between the power measurements and the predetermined value set. We focused on the power trace between the end of the last inner loop and the start of the second outer loop. The Pearson correlation value will be very high at the timestamp when `-neg` was computed.

Figure 4 (top) illustrates the Pearson correlation observed over time for the first leaky operation, `-(t & (f ^ 1))` and the (bottom) second leaky operation, `-neg`. We found that the timestamps with the highest correlation occur at samples 2004 and 3300, corresponding to when `-(t & (f ^ 1))` was computed in the first and third loop execution. For `-neg`, we identified the timestamp with the highest correlation at sample 19460.

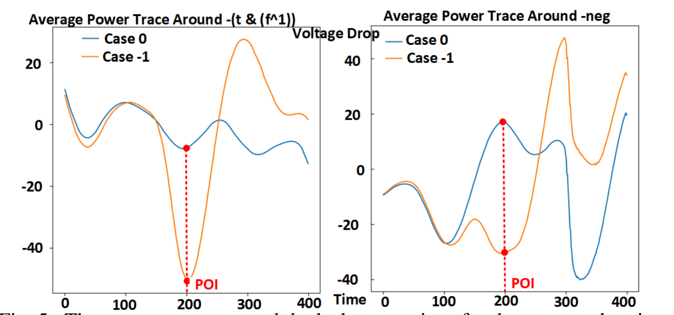

Fig. 5. The average trace around the leaky operations for the two attack points is depicted. The left figure corresponds to the first attack point, while the right figure represents the second attack point. At both attack points, there are only two possible cases: '0' and '-1'. It is observed that the power consumption for case '-1' is higher than for case '0'.

V. Evaluating the Single-Trace Vulnerability

In this section, we first describe our measurement setup, followed by the analysis of the collected side-channel information and presentation of the results. We begin by demonstrating that our attack successfully distinguishes between different intermediate value assignments through graphical visualizations. We then quantify our attack success rate. Specifically, we employed a univariate Gaussian template [34] with the selected point of interest (POI) to quantify the success rate using a theoretical model. Finally, we analyze the impact of the proposed attack on the security of FALCON.

A. Measurement Setup

The target device is an ARM Cortex-M4F CPU operating at 30MHz, which is a canonical setting to test side channels on embedded applications [26], [31]. We utilized the submission package from the NIST reference software implementation. The code was compiled using the `gcc-arm-none-eabi-4_8-2014q1` compiler with `-O0` optimization flag. Measurements were captured with a PicoScope 3206D oscilloscope, set to a sampling rate of 250MHz, using a Tektronix CT1 passive current probe with a bandwidth of 1–1000 MHz at 3 dB. No external amplification was applied to enhance the measurements.

B. Attack Results

1) Results on `-(t & (f ^ 1))`

The left graph in Figure 5 shows the average power trace of the 400 samples around the point of interest (POI) when attacking `-(t & (f ^ 1))`. The horizontal axis represents time in samples, while the vertical axis indicates power consumption. Time sample point 200 marks the point of interest (POI) identified in section IV. Our observations reveal that the average power consumption for case '-1' is higher than for case '0'.

We then quantified the attacker's success rate by modeling the power distribution to derive a theoretical estimate. Since FALCON executes the targeted discrete Gaussian sampling step once to generate a single secret coefficient, we perform a single-trace template attack. We selected the point of highest correlation as the POI to build our univariate Gaussian template, though multiple POIs could be chosen to enhance the success rate on noisier platforms. For this POI, we calculated the average power μ_i and variance of power v_i.

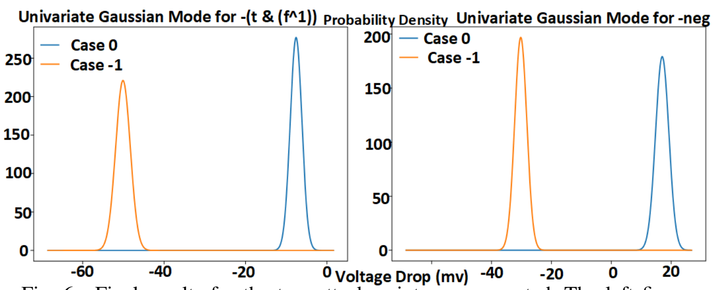

Fig. 6. Final results for the two attack points are presented. The left figure visualizes the result for the first attack point, while the right figure shows the results for the second attack point. We used a univariate Gaussian model. The results are clearly separated in both cases, with no overlap, indicating distinct and successful classification.

$$P_k = \sum_{j=0}^{k} \log \mathcal{N}(t_{j,s_i}, \mu_i, v_i) \quad , \tag{3}$$

We constructed the template using 500k measurements with a normal probability density function (NPDF) based on the POI of the attacked traces t_{j,s_i}, μ_i, and v_i results from the profiling. To avoid precision issues that can arise when the NPDF values are extremely large or small, we summed the logarithms of the normal distribution $\mathcal{N}$ shown in Equation 3. The index of the matrix P_k with the highest value corresponds to the predicted coefficient.

The left figure of Figure 6 illustrates the Gaussian model derived from the data at the POI. The horizontal axis represents power consumption measured by voltage drop, while the vertical axis indicates the probability density. The two bell-shaped curves correspond to the two cases. For this attack point, the results are separated, with overlapping area taking $2.56 \times 10^{-9}\%$ of the total area under the two curves, indicating classification accuracy higher than 99.999999999%. The obtained Gaussian model was applied to 500k collected measurements to derive classification labels. A comparison with the ground truth shows that in practice we didn't see any samples causing a false misidentification.

2) Results on `-neg`

We followed the same template-building approach to evaluate the attack results on `-neg`. The right graph of Figure 5 illustrates the separation between cases '0' and '-1' as revealed by our attack, using average traces. The horizontal axis represents time samples, while the vertical axis reflects power consumption. The results demonstrate a clear distinction on average traces between the two cases.

The right figure of Figure 6 illustrates the Gaussian model obtained from the first step of our attack. We selected the point of highest correlation as the POI to build our univariate Gaussian template. The horizontal axis represents power consumption, and the vertical axis denotes probability density. The two bell-shaped curves illustrate the two cases, with the results separated at this attack point. The overlap accounts for only $2.55 \times 10^{-10}\%$ of the total area under the curves, indicating classification accuracy higher than 99.9999999999%. The Gaussian model was applied to the 500k measurements, and the classification results were compared with the ground truth, achieving a real-world accuracy of 100%.

C. The Effect on FALCON's Security Schemes

Each discrete Gaussian sampling subroutine execution involves running the outer loop twice and running the inner loop 52 times. Our Attack on `-(t & (f ^ 1))` accomplished 99.999999999% accuracy and attack on `-neg` accomplished 99.9999999999% accuracy. Consequently, the overall success rate of our attack to extract one coefficient in FALCON is:

$$(99.999999999\%)^{52} \times (99.9999999999\%)^2 = 99.9999999478\%$$

Providing the success rate above, our overall success rate for inferring the two full secret polynomials, both f and g, for FALCON-512 is:

$$((99.9999999478\%)^{512})^2 = 99.99994654\%$$

For FALCON-1024, Our overall success rate for inferring the two full secret polynomials is:

$$((99.9999999478\%)^{1024})^2 = 99.99989309\%$$

We assert that this vulnerability represents a significant compromise of the FALCON cryptographic scheme.

VI. Discussions

In this section, we discuss defense methods and related issues. We also comment on the drawbacks of our attack.

A. Defense Methods

Defenses against single-trace side-channel vulnerabilities can be implemented at both the hardware and software levels. On the hardware side, constant power consumption hardware designs can mitigate information leakage [35], [36]. On the software side, techniques such as hiding can be implemented, where noise or random delays are introduced to reduce the correlation between power consumption and executed operations [37].

B. Applicability to Other Implementations

Our proposed attack also applies to other algorithms that negate a right-shifted 63-bit value (for 64-bit variables) on secret intermediate variables. The identified vulnerability exists in both FALCON's optimized and reference implementations, as the discrete Gaussian sampling subroutine is the same. Though we implemented our attack on the implementation of FALCON-512, such an attack also efficiently breaks FALCON-1024.

C. Calibration Factors of the Experiments

The platform's noise level decreases as the device's operating frequency is lowered. To reduce noise in our experiments, we set the development board to its minimum frequency of 30 MHz as in prior work [26]. Earlier works have conducted single-trace side-channel attacks at even lower frequencies, such as 8 MHz for attacks on the NTT [16]. Since our clock frequency is higher than in these studies, analyzing higher frequencies may require more sophisticated equipment for power measurement, additional probes for near-field electromagnetic leakage detection, or noise reduction through amplification and post-processing.

D. Drawbacks of Our Attack

Software configurations and peripheral settings could impact the power consumption of the target device. For example, changing the compiler options or flags could change the instruction sequence or add/remove some instructions. Our attack was conducted at the compiler optimization level `-O0`, where we achieved a high attack success rate. We used `-O0` because it preserves the intended structure and sequence of the originally developed code. An exhaustive evaluation of all compiler optimization settings and flags is left for future work.

Template attacks have well-known limitations and challenges, including issues related to cross-device applicability and processing time constraints. In our scenario, we selected a single POI for the attack, which resulted in a reasonable success rate and required a few minutes to build the profiles. Incorporating additional POIs could further enhance the attack's success rate, though it would also demand more processing time.

Our attack was conducted on a single device. For attacks to succeed on devices of different makes and models, it is essential to develop distinct power profiles that account for device-specific features such as pipelining and out-of-order execution. Even for devices of the same make and model, variations arising from manufacturing differences, device aging, and environmental conditions must still be considered. Overcoming these challenges may require advanced machine learning-based profiling techniques [38], [39] or building the template again on each new device under test. It is important to recognize that the challenge of cross-device single-trace side-channel attacks on post-quantum cryptosystems remains an open problem, as noted in several previous studies [16], [27], [20], [23], [40], [41].

VII. Conclusions

Although lattice cryptography is a versatile tool that offers quantum resilience at a reasonable cost, it includes unique operations that have not been thoroughly analyzed for side-channel vulnerabilities. This paper highlights a critical vulnerability in the software implementation of FALCON, specifically in the negation of the right-shift 63-bit operation. We have demonstrated that the discrete Gaussian sampling implementation in FALCON can expose intermediate value assignments, leading to full secret key recovery. Our results confirm the practicality of this attack on an off-the-shelf device. The vulnerability we uncovered is distinct from prior single-trace attacks and, by definition, from multi-trace attacks. As a result, the defenses proposed or implemented for other vulnerabilities must be re-evaluated, as they are likely ineffective against this specific form of leakage. We emphasize that this is an attack paper, and our primary goal is to inform the implementers of these algorithms about the vulnerabilities they introduce.

References

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

[2] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of computation*, vol. 48, no. 177, pp. 203–209, 1987.

[3] J. Proos and C. Zalka, "Shor's discrete logarithm quantum algorithm for elliptic curves," *arXiv preprint quant-ph/0301141*, 2003.

[4] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM review*, vol. 41, no. 2, pp. 303–332, 1999.

[5] L. Chen, L. Chen, S. Jordan, Y.-K. Liu, D. Moody, R. Peralta, R. Perlner, and D. Smith-Tone, *Report on post-quantum cryptography*. US Department of Commerce, National Institute of Standards and Technology, 2016, vol. 12.

[6] ——, *Report on post-quantum cryptography*. US Department of Commerce, National Institute of Standards and Technology, 2016, vol. 12.

[7] V. Lyubashevsky, L. Ducas, E. Kiltz, T. Lepoint, P. Schwabe, G. Seiler, D. Stehlé, and S. Bai, "Crystals-dilithium," *Algorithm Specifications and Supporting Documentation*, 2020.

[8] D. J. Bernstein, A. Hülsing, S. Kölbl, R. Niederhagen, J. Rijneveld, and P. Schwabe, "The sphincs+ signature framework," in *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, 2019, pp. 2129–2146.

[9] T. Prest, P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Ricosset, G. Seiler, W. Whyte, and Z. Zhang, "Falcon," *Post-Quantum Cryptography Project of NIST*, 2020.

[10] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Annual international cryptology conference*. Springer, 1999, pp. 388–397.

[11] P. Ravi, S. S. Roy, A. Chattopadhyay, and S. Bhasin, "Generic side-channel attacks on cca-secure lattice-based pke and kems," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 307–335, 2020.

[12] K. Ngo, E. Dubrova, Q. Guo, and T. Johansson, "A side-channel attack on a masked ind-cca secure saber kem."

[13] A. Sayakkara, N.-A. Le-Khac, and M. Scanlon, "A survey of electromagnetic side-channel attacks and discussion on their case-progressing potential for digital forensics," *Digital Investigation*, vol. 29, pp. 43–54, 2019.

[14] T. Tosun and E. Savas, "Zero-value filtering for accelerating non-profiled side-channel attack on incomplete ntt-based implementations of lattice-based cryptography," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 3353–3365, 2024.

[15] R. Primas, P. Pessl, and S. Mangard, "Single-trace side-channel attacks on masked lattice-based encryption," in *International Conference on Cryptographic Hardware and Embedded Systems*. Springer, 2017, pp. 513–533.

[16] P. Pessl and R. Primas, "More practical single-trace attacks on the number theoretic transform," in *International Conference on Cryptology and Information Security in Latin America*. Springer, 2019, pp. 130–149.

[17] I.-J. Kim, T.-H. Lee, J. Han, B.-Y. Sim, and D.-G. Han, "Novel single-trace ml profiling attacks on nist 3 round candidate dilithium," 2020.

[18] T. Espitau, P.-A. Fouque, B. Gérard, and M. Tibouchi, "Side-channel attacks on bliss lattice-based signatures: Exploiting branch tracing against strongswan and electromagnetic emanations in microcontrollers," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1857–1874.

[19] 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 Symposium on Hardware Oriented Security and Trust (HOST)*, 2018, pp. 81–88.

[20] J. W. Bos, S. Friedberger, M. Martinoli, E. Oswald, and M. Stam, "Assessing the feasibility of single trace power analysis of frodo," in *International Conference on Selected Areas in Cryptography*. Springer, 2018, pp. 216–234.

[21] Z. Qiao, Y. Liu, Y. Zhou, Y. Zhao, and S. Chen, "Single trace is all it takes: Efficient side-channel attack on dilithium," *Cryptology ePrint Archive*, 2024.

[22] B.-Y. Sim, J. Kwon, J. Lee, I.-J. Kim, T. Lee, J. Han, H. Yoon, J. Cho, and D.-G. Han, "Single-trace attacks on the message encoding of lattice-based kems," Cryptology ePrint Archive, Report 2020/992, 2020, https://eprint.iacr.org/2020/992.

[23] D. Amiet, A. Curiger, L. Leuenberger, and P. Zbinden, "Defeating newhope with a single trace," in *International Conference on Post-Quantum Cryptography*. Springer, 2020, pp. 189–205.

[24] Z. Huang, H. Wang, B. Cao, D. He, and J. Wang, "A comprehensive side-channel leakage assessment of crystals-kyber in iiot," *Internet of Things*, vol. 27, p. 101331, 2024.

[25] T. Rabas, J. Buček, and R. Lórencz, "Single-trace side-channel attacks on ntru implementation," *SN Computer Science*, vol. 5, no. 2, p. 239, 2024.

[26] E. Karabulut, E. Alkim, and A. Aysu, "Single-trace side-channel attacks on ω-small polynomial sampling: with applications to ntru, ntru prime, and crystals-dilithium," in *2021 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. IEEE, 2021, pp. 35–45.

[27] S. Kim and S. Hong, "Single trace analysis on constant time cdt sampler and its countermeasure," *Applied Sciences*, vol. 8, no. 10, p. 1809, 2018.

[28] K.-H. Choi, J. Han, and D.-G. Han, "Single trace analysis of visible vs. invisible leakage for comparison operation based cdt sampling," 2024.

[29] S. Jendral, K. Ngo, R. Wang, and E. Dubrova, "Breaking sca-protected crystals-kyber with a single trace," in *2024 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. IEEE, 2024, pp. 70–73.

[30] E. Karabulut and A. Aysu, "Falcon down: Breaking falcon post-quantum signature scheme through side-channel attacks," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2021, pp. 691–696.

[31] S. Zhang, X. Lin, Y. Yu, and W. Wang, "Improved power analysis attacks on falcon," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2023, pp. 565–595.

[32] S. McCarthy, J. Howe, N. Smyth, S. Brannigan, and M. O'Neill, "Bearz attack falcon: implementation attacks with countermeasures on the falcon signature scheme," *Cryptology ePrint Archive*, 2019.

[33] M. Guerreau, A. Martinelli, T. Ricosset, and M. Rossi, "The hidden parallelepiped is back again: Power analysis attacks on falcon," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 141–164, 2022.

[34] S. Chari, J. R. Rao, and P. Rohatgi, "Template attacks," in *International Workshop on Cryptographic Hardware and Embedded Systems*. Springer, 2002, pp. 13–28.

[35] K. Tiri and I. Verbauwhede, "Design method for constant power consumption of differential logic circuits," in *Design, Automation and Test in Europe*, 2005, pp. 628–633 Vol. 1.

[36] X. Lou, T. Zhang, J. Jiang, and Y. Zhang, "A survey of microarchitectural side-channel vulnerabilities, attacks and defenses in cryptography," *CoRR*, vol. abs/2103.14244, 2021. [Online]. Available: https://arxiv.org/abs/2103.14244

[37] M. Brisfors, M. Moraitis, and E. Dubrova, "Side-channel attack countermeasures based on clock randomization have a fundamental flaw," Cryptology ePrint Archive, Paper 2022/1416, 2022. [Online]. Available: https://eprint.iacr.org/2022/1416

[38] P. Kashyap, F. Aydin, S. Potluri, P. Franzon, and A. Aysu, "2deep: Enhancing side-channel attacks on lattice-based key-exchange via 2d deep learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2020.

[39] J. Kim, S. Picek, A. Heuser, S. Bhasin, and A. Hanjalic, "Make some noise. unleashing the power of convolutional neural networks for profiled side-channel analysis," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 148–179, 2019.

[40] C. Zhang, Z. Liu, Y. Chen, J. Lu, and D. Liu, "A flexible and generic gaussian sampler with power side-channel countermeasures for quantum-secure internet of things," *IEEE Internet of Things Journal*, 2020.

[41] 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.