# Single-trace clustering power analysis of the point-swapping procedure in the three point ladder of Cortex-M4 SIKE

Aymeric Genêt[1,2] and Novak Kaluđerović[1]

[1] École Polytechnique Fédérale de Lausanne, Ecublens, Switzerland
{aymeric.genet, novak.kaluderovic}@epfl.ch
[2] Kudelski Group, Cheseaux-sur-Lausanne, Switzerland
aymeric.genet@nagra.com

**Abstract.** In this paper, the recommended implementation of the post-quantum key exchange SIKE for Cortex-M4 is attacked through power analysis with a single trace by clustering with the $k$-means algorithm the power samples of all the invocations of the elliptic curve point swapping function in the constant-time coordinate-randomized three point ladder. Because each sample depends on whether two consecutive bits of the private key are the same or not, a successful clustering (with $k = 2$) leads to the recovery of the entire private key. The attack is naturally improved with better strategies, such as clustering the samples in the frequency domain or processing the traces with a wavelet transform, using a simpler clustering algorithm based on thresholding, and using metrics to prioritize certain keys for key validation. The attack and the proposed improvements were experimentally verified using the ChipWhisperer framework. Splitting the swapping mask into multiple shares is suggested as an effective countermeasure.

**Keywords:** SIKE · side-channel analysis · power analysis · $k$-means clustering · single-trace attack · post-quantum key exchange · isogeny-based cryptography · ARM cortex-m4

## 1 Introduction

The cryptographic world as we know it will be coming to an end once a big enough quantum computer is stabilized. Such a prophecy was foretold due to the quantum capability of Shor's algorithm [49] to factor large numbers in polynomial time. As a result, cryptographers are currently developing substitutes for cryptographic primitives that resist quantum computation based attacks (known as *post-quantum cryptography*).

To further address the risk, the NIST (National Institute of Standards and Technology) supervises a standardization process for post-quantum cryptography [39]. The project aims to standardize the solutions that were submitted by cryptographers in a round-based competition-like process. Currently, the process

reached its third round which comprises finalists (four key exchange schemes and three digital signature schemes), as well as alternate candidates that are retained but currently lack the sufficient scrutiny to advance further in the process [40].

Among the schemes that reached the third round, SIKE [29] (Supersingular Isogeny Key Exchange) is listed as an alternate candidate. SIKE is a key exchange mechanism which bases its security on the difficulty to compute isogenies between two elliptic curves. The key exchange works in a Diffie–Hellman fashion, where two parties (Alice and Bob) walk in a supersingular isogeny graph and exchange concealed information to reach a common curve. While SIKE is a relatively novel cryptosystem, the scheme still operates with methods from classical Elliptic Curve Cryptography (ECC), such as the scalar multiplication of elliptic curve points using a variant of the Montgomery ladder with three points (the *three point ladder*).

SIKE is, even though quantum-resistant, still a classical cryptosystem running on classical electronic devices. As these devices are subject to additional vectors of attacks, such as *side-channel attacks* in which secret information is recovered by measuring physical variables, SIKE can still be vulnerable when implemented in practice.

Power consumption is an example of such a leaking physical variable; leading thus to *side-channel power analysis* [30]. This analysis considers an adversary who typically has physical access to the targeted device and is therefore able to measure *power traces*; a collection of instantaneous samples of power consumption related to the executions of the cryptographic algorithms. The adversary's goal is to exploit the link between power consumption and data processed to infer information about the secrets concealed within the device from the measured power traces.

There exist many different ways of taking advantage of power consumption measurements in a side-channel power analysis. The most straightforward is to identify secret-dependent patterns in a power trace; an example of a Simple Power Analysis (SPA) [30,31]. Others, more complex, attempt to model the power consumption in order to apply statistical techniques, such as Correlation Power Analysis (CPA) [9]. Machine learning algorithms [26,33] can be trained to retrieve the secret values from the power consumption but usually necessitate a programmable device that is identical to the target device. This is avoided with unsupervised learning algorithms—in particular, clustering algorithms [43]—that are able to learn about secret values without creating a profile of the cryptographic device. Recently, deep learning algorithms have been used on top of unsupervised clustering to significantly increase the success of the power analysis [42].

### 1.1 Related work

The first paper to ever consider clustering algorithms in side-channel analysis is due to Heyszl et al. [27] who used the $k$-means clustering algorithm to recover the private-key bits of an elliptic curve scalar multiplication, but on power samples from simultaneous electromagnetic probes positioned on different locations on

a chip to boost the independent recovery of bits. Perin et al. extended this approach to a single-probe in [43] by combining the classification of multiple private-key bits in a fully-unsupervised process which was experimentally verified on a protected implementation of RSA. The attack was then further improved by Specht et al. in [52] by pre-processing the traces with a principal component analysis and by using expectation-maximization as a clustering algorithm. Then, Perin and Chmielewski proposed a semi-parametric framework that uses non-profiled learning both as a leakage assessment tool and as a private-key bits recovery tool in [41]. A practical unsupervised attack against ECC was mounted on the protected library of $\mu$NaCl in [37]. Recently, several other libraries were attacked with clustering power analysis on smartphones with low-cost equipment in [2]. Independent studies which also investigated clustering power analysis but on different parts of the Montgomery ladder obtained similar results in [50,48]. Finally, in [42], Perin et al. established the current state of the art in clustering power analysis by using deep learning methods on the traces and the output clusters to predict and correct wrong labelings.

Single-trace attacks against ECC are not limited to clustering power analysis. Horizontal correlation power analysis [12,44], for instance, can sequentially recover bits of a private key by correlating hypothesized intermediate values on the power traces. Other notable single-trace analyses are horizontal collision attacks [57,16] (also known as "Big Mac" attacks) which work by comparing a portion of the power trace with another to distinguish whether the processed data were similar or dissimilar. Template attacks [10,17,36,38] recover noisy information about specific secret values by operating in two phases: a profile of the power consumption that corresponds to (known) secret values is first created, and then applied to the power consumption of a target device. Online template attacks [6,19,3] propose an adaptative model in which profiles of the power consumption are created *after* receiving the target power trace. Templates of many different intermediate values can also be combined with belief propagation [56,4] to improve the recovery of said values.

Regarding the scrutiny of side-channel analysis on isogeny-based cryptography, the first concerns date back to 2017 [32], but the first practical side-channel attacks on SIKE were carried out in 2020 by Zhang et al. in [58] where the authors apply a multiple-trace differential power analysis; requiring thus one of the two parties key to be fixed. This study was extended to a single-trace horizontal correlation power analysis in [25] where the key pairs of both parties can be ephemeral. In parallel, isogeny-based cryptography was investigated with fault injections with a loop-abort attack in [24], and a corruption of the auxiliary points in [54]. The latter was verified experimentally in [53].

### 1.2 Contributions

This paper presents a side-channel clustering power analysis against the SIKE implementation of [47] which is recommended for the ARM Cortex-M4 microcontroller on the official SIKE website [29].

The attack works by clustering combined power samples of the calls to the `swap_points` function in the coordinate-randomized three point ladder involved in key exchange. Since the function swaps multiple values only when two consecutive private bits are different, the samples are expected to follow two distinct distributions (i.e., the values were swapped or not). If these sample distributions are distinct enough, the labeling of the provided measurements will directly lead to the bits of the private key thus compromising the confidentiality guarantees of the key exchange. Note that the attack is not limited to SIKE nor to the three point ladder, but to all swapping procedures based on masking values.

The most important feature of the attack is that a *single* execution of the key exchange is sufficient, and that only samples from the power domain of the microcontroller are necessary. More specifically, neither electromagnetic sampling, nor profiling, nor previous knowledge of the target device is required. The key recovery is completely *non*-supervised (i.e., no template is ever built) and can be performed without the need for public keys (which are still required to verify whether the key was successfully extracted).

As a result, since the attack exploits power consumption rather than localized electromagnetic radiations, the attack does not target specific registers and is expected to defeat countermeasures based on randomizing the physical locations of the registers, as initially studied by Heyszl et al. in [28] and recommended as an effective countermeasure in [37]. Moreover, in comparison with the approach of Nascimento and Chmielewski from [37], the attack is shown to successfully extract the key without, in particular, requiring leakage assessment, even though their approach is still expected to fully work on the same target.

In addition to the previous improvements, the paper exhibits the following contributions:

- Clustering the samples is shown to be also possible in the frequency domain, i.e., by first processing the power traces with a Fourier transform. As a result, the attack becomes tolerant to cases where traces are slightly misaligned due to, e.g., bad segmentation.
- Alternatively, the clustering power analysis can be improved by processing the traces with a wavelet transform to compress the power traces by filtering out high frequencies. Such a transform is shown to be relevant in practice because the compression reduces the number of timing locations to visit.
- A clustering method based on thresholding the distribution of sorted power samples is shown to be sufficient to successfully recover the key. This result shows that the clustering algorithm does not require to be sophisticated, and can be far less complex than the methods proposed so far.
- Finally, a countermeasure based on splitting the masking value into multiple random shares during the swapping procedure is shown to effectively protect against the attack described in the paper.

### 1.3 Outline

Section 2 briefly recalls SIKE and the $k$-means clustering algorithm. The clustering power analysis is then presented in Section 3, which is followed by further

enhancements to improve the efficiency of the attack in Section 4. The experimental verification of the attack and improvements are explained in Section 5. Then, in Section 6, the suggested countermeasure is described and validated so that the paper can finally conclude with Section 7.

## 2 Background

### 2.1 Supersingular Isogeny Key Encapsulation (SIKE)

The following sub-section gives a brief overview of the SIKE protocol with the emphasis being put on the three point ladder. A reader interested in a full description of SIKE is advised to read [29,22].

| Alice | | Bob |
|---|---|---|
| | | $sk_B \leftarrow_\$ [0, 3^{e_B})$ |
| | | $s \leftarrow_\$ \{0,1\}^\tau$ |
| | | $R_B \leftarrow P_B + [sk_B]Q_B$ |
| | | Let $\phi_B : E_0 \rightarrow E_B$ be s.t. $\ker(\phi_B) = \langle R_B \rangle$ |
| | $\xleftarrow{pk_B}$ | $pk_B = (E_B, \phi_B(P_A), \phi_B(Q_A))$ |
| $m \leftarrow_\$ \{0,1\}^\tau$ | | |
| $sk_A \leftarrow G(m\|pk_B) \bmod 2^{e_A}$ | | |
| $R_A \leftarrow P_A + [sk_A]Q_A$ | | |
| Let $\phi_A : E_0 \rightarrow E_A$ be s.t. $\ker(\phi_A) = \langle R_A \rangle$ | | |
| $R'_A \leftarrow \phi_B(P_A) + [sk_A]\phi_B(Q_A)$ | | |
| Let $\phi'_A : E_B \rightarrow E_{AB}$ be s.t. $\ker(\phi'_A) = \langle R_A \rangle$ | | |
| $c_0 = (E_A, \phi_A(P_B), \phi_A(Q_B))$ | | |
| $c_1 = F(j(E_{AB})) \oplus m$ | | |
| $K = H(m\|c_0\|c_1)$ | | |
| | $\xrightarrow{(c_0\|c_1)}$ | $R'_B \leftarrow \phi_A(P_B) + [sk_B]\phi_A(Q_B)$ |
| | | Let $\phi_B : E_B \rightarrow E'_{AB}$ be s.t. $\ker(\phi_B) = \langle R_B \rangle$ |
| | | $m' \leftarrow F(j(E'_{AB})) \oplus c_1$ |
| | | $sk'_A \leftarrow G(m'\|pk_B) \bmod 2^{e_A}$ |
| | | $R' \leftarrow P_A + [sk'_A]Q_A$ |
| | | Let $\phi' : E_0 \rightarrow E'_A$ be s.t. $\ker(\phi') = \langle R' \rangle$ |
| | | **if** $(E'_A, \phi'(P_B), \phi'(Q_B)) = c_0$ |
| | | $K = H(m'\|c_0\|c_1)$ |
| | | **else** |
| | | $K = H(s\|c_0\|c_1)$ |

Fig. 1: The SIKE protocol.

**Protocol summary.** SIKE [29] is an isogeny-based key exchange mechanism which extends SIDH [22] with the Fujisaki–Okamoto transform [23].

The public parameters of the protocol include a length $\tau > 0$, a prime $p = 2^{e_A}3^{e_B} - 1$ (s.t. $2^{e_A} \approx 3^{e_B}$), a starting elliptic curve: $E_0 = \{(x, y) \in \mathbf{F}_{p^2} : y^2 = x^3 + 6x^2 + x\}$, two basis points $(P_A, Q_A)$ of the $2^{e_A}$-torsion, two basis points $(P_B, Q_B)$ of the $3^{e_B}$-torsion, and three cryptographic hash functions $F$, $G$, and $H$.

The protocol works as shown in Fig. 1. The isogenies $\phi_A$, $\phi_B$ can be computed using Vélu's formula [55]. The function $j(E)$ returns the $j$-invariant of the elliptic curve $E$.

**Three point ladder.** SIKE makes use of the *three point ladder* [21] to compute a secret point that is then used to generate the kernel of a party's secret isogeny. More specifically, the three point ladder is an optimized computation of the elliptic curve operation of $P + [sk]Q$ for a scalar $sk$ of $n$ bits, by using only the $X$ (and $Z$) coordinates of the points $Q$, $P$, and $Q - P$ on a Montgomery curve.

Concretely, the three point ladder uses three variables: `R0`, `R`, and `R2`, whose values respectively start with both the $X$ and $Z$ coordinates of $Q$, $P$, and $Q-P$. Then, the bits of the private scalar $sk$ are scanned from the least to the most significant bit. For each bit, `R0` is either added to `R` if the bit is one, or to `R2` if the bit is zero. The point `R0` is always doubled at the end of each iteration.

*Implementation.* The implementation of the three point ladder under consideration is the one from [47] (shown in Appendix A, Listing 1.1). The main feature of this implementation is that the point `R` is swapped with `R2` depending on the difference between the current and the previous private bits, so `R0` can always be added to `R2`.

*Swapping procedure.* To perform the conditional swapping of points, a function `swap_points` (also shown in Appendix A, Listing 1.2) takes `R` and `R2` as parameters, as well as a mask that expands the value of the private bit difference on a whole word. Given two consecutive private-key bits $sk_{i-1}, sk_i$ (for $0 \leq i < n$ with $sk_{-1} = sk_n = 0$), with 32-bit words, such a mask corresponds to:

$$\texttt{mask} = \begin{cases} \texttt{0x00000000} & \text{if } sk_{i-1} \oplus sk_i = 0, \\ \texttt{0xFFFFFFFF} & \text{if } sk_{i-1} \oplus sk_i = 1. \end{cases}$$

Then, each word of the two elliptic curve points (resp. $a$ and $b$) are processed according to this formula:

$$\begin{aligned} \texttt{tmp} &= \texttt{mask} \mathbin{\&} (a \oplus b), \\ a &= \texttt{tmp} \oplus a, \\ b &= \texttt{tmp} \oplus b, \end{aligned}$$

where & corresponds to the bitwise "and" operator. Such a procedure is known to achieve constancy in timing and execution (see [15]).

*Coordinate randomization.* To prevent an adversary from exploiting the values of the points in a power analysis, a random non-zero field element $r$ can be

uniformly drawn and multiplied into the $X$ and $Z$ coordinates [13]. This is based on the observation that, in projective coordinates, an elliptic curve point $(X : Y : Z)$ is equivalent to $(rX : rY : rZ)$ (for $r \neq 0$). This countermeasure requires $3 \times \log_2(p^2) \approx 12e_A$ bits of entropy and six additional field multiplications per iteration.

### 2.2 Clustering

**k-means.** The *k-means algorithm* [34] is an unsupervised clustering algorithm that partitions a population of $n$ samples[3] into $k$ sets solely based on the values of the samples.

Informally, the algorithm starts with $k$ groups of means $\mu_j$ $(0 \leq j < k)$, and reassigns the samples to the group with the closest mean. As doing so may change the means of the groups, the process is repeated until convergence. The procedure is shown in pseudocode in Algorithm 1.

**Algorithm 1** The $k$-means algorithm.

**Require:** $\{s_i \in \mathbf{R}\}_{0 \leq i < n}$: Collection of $n$ samples.
1: Assign each $s_i$ to a cluster $j$ at random $(0 \leq i < n,\ 0 \leq j < k)$.
2: **repeat**
3: Compute each $\mu_j$ as the mean of each cluster $(0 \leq j < k)$.
4: Assign each $s_i$ to the cluster $j = \text{argmin}|s_i - \mu_j|$ $(0 \leq i < n)$.
5: **until** no $\mu_j$ change (for all $0 \leq j < k$).
6: **return** the final cluster assignments of all $s_i$ $(0 \leq i < n)$.

## 3 Clustering Power Analysis of SIKE

This section describes the attack which recovers a party's private key by classifying the power samples (in Volts) of a single execution of the three point ladder in SIKE.

*Target.* The attack targets the three point ladder of $n$ bits as described in Section 2.1. As a result, the attack can be applied at every stage of the protocol. For the sake of simplicity, the paper assumes that the attack targets the three point ladder invoked in the key decapsulation.

*Threat model.* The attack assumes a passive adversary able to monitor the messages exchanged in the SIKE protocol, and the power consumption of the attacked device.

*Traces collection.* In the premise of the attack, the power consumption of the entire three point ladder is measured with a fixed and fast enough sampling rate. The power samples are then segmented into multiple power traces synchronized at the beginning of each step of the three point ladder. Moreover, only

[3] In the scope of this paper, the population is one-dimensional.

**Algorithm 2** Clustering power analysis.

**Require:** $\{s_i\}_{0 \leq i < n}$: Collection of all the power samples at a same $t$.

1: Run the $k$-means clustering on $\{s_i\}_{0 \leq i < n}$ (with $k = 2$).
2: Let $sk_{-1} = 0$.
3: Let $l_i = \begin{cases} 0 \text{ if } s_i \in \text{first cluster,} \\ 1 \text{ if } s_i \in \text{second cluster,} \end{cases}$ $(i \leq 0 < n)$.
4: Let $sk_i = l_i \oplus sk_{i-1}$ $(i \leq 0 < n)$.
5: **return** $sk = (sk_0, sk_1, sk_2, \ldots, sk_{n-2}, sk_{n-1})$.

the segments corresponding to the execution of the `swap_points` functions are considered, each of them ultimately consisting of $M$ samples (typically, a few thousands).

*Rationale.* Because `swap_points` masks values with either `0x00000000` or `0xFFFFFFFF` depending on the difference between two consecutive secret bits, the attack attempts to distinguish for each iteration whether the swap occurred or not by gathering the samples at a same location in *all* iterations and clustering them with $k$-means. Since a difference of bits can only be zero (identical) or one (different), only two clusters are considered (i.e., $k = 2$). The private key can be entirely reconstructed from the labels at the end of the clustering.

*Attack procedure.* Given the $n$ segmented power traces $T_i$ of $M$ power samples each, the procedure consists of three steps:

1. Select a sample location $0 \leq t < M$ in the power traces.
2. Cluster with $k$-means the $n$ power samples at location $t$ throughout the traces $T_i$ (for $0 \leq i < n$), and reconstruct the key from the labels.
3. Verify the key obtained.

Algorithm 2 shows the second step of the attack in more details. The samples $s_i$ must all correspond to the samples at a same time throughout the $n$ segmented power traces. Since Algorithm 2 considers samples from a single timing location in the power traces, the procedure can be repeated with all different positions until a returned key (or its bitwise inverse) successfully decrypts a ciphertext. As a result, the overall attack has a complexity of precisely $M$ executions of $k$-means and key tryouts.

## 4 Attack Enhancements

In a full attack as described in Section 3, the adversary needs to pass through all sample locations in the traces and use $k$-means to recover a deterministic key candidate that eventually needs to be verified. Adopting a better strategy for any of these steps can lead to both faster and more successful results.

This section lists enhancements to speed up the eventual recovery of the key. These can sometimes be combined to improve even further the overall attack.

### 4.1 Enhancing sample selection

The original attack exploits the raw power consumption which requires visiting *all* sample locations. Applying a transform to the power consumption before the clustering step can reduce the number of samples and therefore speed up the attack. Moreover, relating the power consumption to a different domain can exhibit leakage points which may lead to improved results.

**Fourier transform.** A (discrete) *Fourier transform* is a decomposition of a (discrete) signal into a representation that exhibits information about the frequency components of the signal. The representation is obtained by projecting the signal $s_i$ $(0 \leq i < M)$ onto the (discrete) orthogonal Fourier basis $\{e^{I2\pi if/M}\}_{0 \leq f < M} \in \mathbf{C}^M$ (where $I = \sqrt{-1}$ is the imaginary unit):

$$\hat{s}_f = \sum_{i=0}^{M-1} s_i e^{-I2\pi if/M} \qquad (0 \leq f < M).$$

In power analysis, the signal corresponds to the power consumption, and the frequency coefficients are associated to the samples from operations being performed at fixed intervals. Exploiting the power leakages in the frequency domain is a well-studied process (see [1]).

In the attack, because the power trace captures the operations that periodically swap words in a regular for-loop, some of the frequency coefficients are also expected to follow two distinct distributions. As a result, processing the power consumption with a Fourier transform prior to running the clustering algorithm is expected to give a similar success rate.

Clustering in the Fourier domain exhibits many advantages over clustering in the time domain:

- Due to the Hermitian symmetry ($\hat{s}_f = \hat{s}^*_{-f}$), the upper half of the coefficients is identical to the lower half. Accordingly, the number of locations visited by the clustering algorithm can be divided by two. Moreover, this number can be reduced by only considering a range of reasonable frequencies (such as all the frequencies below the clock speed of the targeted device).
- As the Fourier analysis treats the frequency components of the signal, the processed signal is tolerant to timing misalignments. Such misalignments are particularly common when monitoring the power consumption for a long time, or when segmenting a long power trace.
- The frequencies of interest (i.e., frequencies at which the information leakage is significant) are expected to be unique to a single device and can therefore be re-used in a subsequent analysis (resulting in an educated but still unsupervised attack).

**Wavelet transform.** A (discrete) *wavelet transform* is a multi-level filter bank parameterized by a wavelet function $\psi(t)$ which decomposes a (discrete) signal

into frequency bands. As opposed to the Fourier transform, the wavelet transform gathers information both from the frequency and the timing contents by iteratively correlating the signal with $\{1/\sqrt{2^j}\psi((t-2^j i)/2^j)\}_{(i,j)\in\mathbf{Z}^2}$.

A single step of the filter bank separates an input signal $s_i$ $(0 \leq i < M)$ into two sub-signals of respectively low and high frequencies:

1) the *approximations*: $a_i = \sum_{k=-\infty}^{\infty} s_f L_{2i-k}$
2) the *details*: $d_i = \sum_{k=-\infty}^{\infty} s_f H_{2i-k}$

where $L_i$ and $H_i$ are respectively low-pass and high-pass filters obtained from the wavelet function $\psi(t)$ (see [35] for the technical details). The filter bank consists of recursing the above formulas with $a_i$ with different scales (i.e., with $2^j$ for $j \geq 0$).

An example of a generic three-level wavelet transform is shown on Fig. 2. Given $f$ the frequency of $s_i$ and $\ell \geq 0$, $a_i^{(\ell)}$ corresponds to the sub-signal of frequencies $[0, f/2^{\ell+1}]$ and is re-injected into the filters $H_i$ and $L_i$ to ultimately output $a_i^{(2)}$, while $d_i^{(\ell)}$ corresponds to the sub-signals of frequencies $[f/2^{\ell+1}, f/2^{\ell}]$. Note that $d_i^{(0)}$, $d_i^{(1)}$, $d_i^{(2)}$, and $a_i^{(2)}$ cover the entire spectrum of $[0, f]$.

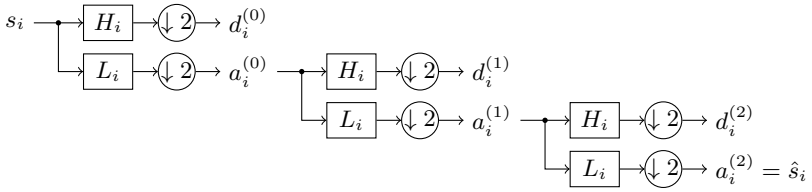



Fig. 2: A three-level wavelet transform.

In power analysis, the wavelet transform is recognized to refine the quality of the power traces acquired (see [11,51]). In clustering power analysis, two advantages are mainly capitalized upon:

- The wavelet transform downsamples the signal at each level while keeping the lower frequency bands. This process halves the length of a power trace each time and therefore results in fewer timing locations to check in the attack.
- By filtering out higher frequencies, the wavelet transform acts as a post-processing de-noiser. Cleaner signals are therefore expected to be output, which anticipates better results.

**Other transforms.** In addition to the Fourier and the wavelet transforms, other transforms that compress the power traces can reduce their number of samples. For instance, principal component analysis [8] is a technique that reduces the

dimensionality of the power traces. Such a transform was also reported to obtain better results in [52] by selecting the significant principal components.

### 4.2 Enhancing power samples clustering

Since the overall attack needs to run a clustering algorithm several times, an algorithm that clusters the power samples more efficiently leads to faster results.

**Thresholding.** While Algorithm 1 already involves low-complexity computations, the clustering algorithm does not needs to be generic and can therefore be tailored to a one-dimensional two-population problem by splitting the distributions with an appropriate middle point.

Many solutions exist to find a suitable middle point, such as computing the overall mean of all the samples, or finding the biggest gap between two neighboring power sample. Algorithm 3 proposes a clustering which calculates the literal middle point of the distribution by finding the maximum and minimum. Such a solution runs in $\mathcal{O}(n)$, but can be tweaked to present other advantages that are described in the next subsection.

**Algorithm 3** Thresholding clustering algorithm.

**Require:** $\{s_i \in \mathbf{R}\}_{0 \leq i < n}$: Set of $n$ samples at a same time $t$.
1: Compute $d = (\min(\{s_i\}_{0 \leq i < n}) + \max(\{s_i\}_{0 \leq i < n}))/2$.
2: Let $l_i = \begin{cases} 0 \text{ if } s_i < d, \\ 1 \text{ otherwise,} \end{cases} \quad (0 \leq i < n)$.
3: **return** $(l_i)_{0 \leq i < n}$.

**Other clustering methods.** In a noisy environment, rigid clustering methods such as $k$-means, thresholding, and even expectation-maximization (as used in [52]) are inadequate due to the two clusters overlapping with each other. Relaxed clustering techniques, such as fuzzy $c$-means [20], have been reported to successfully overcome these limitations in [37,43].

### 4.3 Enhancing key verification

The attack achieves a better performance by reducing the number of key candidates to verify, or by correcting plausible clustering mistakes.

**Majority rule.** As noted by [43], a same labeling re-occurring throughout many different locations is likely to be correct. Two majority rules are therefore proposed:

1. A *vertical* majority in which a candidate key occurring multiple times across the timing locations is verified in priority.

2. A *horizontal* majority in which individual key bits are labeled given their majority throughout the clusterings at all timing locations.

In a horizontal majority rule, a threshold can be selected to filter all the bits for which the clusterings give the same results, while the remaining bits can simply be guessed.

**Educated thresholding.** In the clustering power analysis against the three point ladder of SIKE, two observations can be made:

1. A clustering is successful only when the two sub-distributions are distinct.
2. The number of swaps must always be even.

The first observation stems from the fact that two samples of identical value should always be assigned to the same cluster. Hence, a successful clustering can only be found by splitting the overall distribution in two in between two sample values.

The second observation is due to the fact that the three point ladder requires the points to always be "unswapped" at the end of the procedure. This means that the sizes of the two clusters are always even which can therefore be used to validate the key found.

As a result, one can design a thresholding algorithm similar to Algorithm 3 that first sorts the power samples and then separates the distribution in two, each call at a different threshold starting from a middle point. The threshold can move depending on the distance between the current threshold and the two cluster centers (similarly as in $k$-means). By iteratively calling such an algorithm, the labels that are more likely to be erroneous can be marked and subsequently flipped in a way that make sure that the Hamming weight of the labeling bitstring is even. The complexity of this new method is $\mathcal{O}(n \log n)$.

**Other post-processing.** In case the sample location is known to correspond to a leakage point but the environment is too noisy to perfectly separate the clusters in two (see [37,43] for context), methods based on deep learning can still successfully extract the key, as reported by Perin et al. in [42].

## 5 Experimental Verification

This section reports a proof of concept for the clustering power analysis described in Section 3, in addition to an evaluation of the efficiency of the enhancements proposed in Section 4.

### 5.1 Setup

**Hardware.** The following equipment was used:

– A common laptop running Linux 5.13.5-arch1-1.

- The ChipWhisperer-Lite Level 2 Starter Kit:
  - A programmable STM32F3 (with a Cortex-M4 clocked at 7.37 MHz).
  - A ChipWhisperer-Lite.
  - A ChipWhisperer's "UFO" board.
  - A 20dB Low-Noise Amplifier (LNA).
- A digital oscilloscope with the following characteristics:
  - A resolution of 10 bits.
  - A bandwidth of 20 MHz.
  - A sampling rate of 250 samples per $\mu$s.
  - A memory of 25,000 samples.

The laptop communicates to the toolkit through a micro-USB cable connected to the ChipWhisperer-Lite, which is itself linked to the "UFO" board through a 20-pin cable. The "UFO" board is connected via its `SHUNTL` port[4] to the oscilloscope through the LNA with optical fibers. The STM32F3 is plugged onto the "UFO" as a shield.

Note also that the sampling rate was intentionally made high to showcase the efficiency of the preprocessing transforms.

**Software.** The software considered is the SIKE implementation for Cortex-M4 of [47] which needs to be used in a certain way that enables power trace collection. In particular, the program that runs on the STM32F3 waits for the laptop to send the three byte-encoded elliptic curve points (i.e., Alice's public key) through a serial communication with the ChipWhisperer-Lite. Such a transfer prepares the STM32F3 to run the three point ladder with a pre-programmed private key which can be triggered anytime.

*Coordinate randomization.* As the three point ladder from [47] does not originally offer protections against power analysis, coordinate randomization was only simulated. In this simulation, three multiplicative field elements are pseudo-randomly generated from a pre-programmed seed at the beginning of each iteration. Since only a cheap pseudo-random number generator was required, ChaCha8 [7] was chosen for this purpose. These pseudo-random elements are respectively multiplied (in $\mathbf{F}_{p^2}$) to the $X$ and $Z$ coordinates of the three points.

Note that while this simulation sufficiently protects the three point ladder from correlation attacks based on the values of the elliptic points (see, e.g., [58,25]), the resulting code is not claimed to be secure in a real-life scenario. This implementation is evidently not what was considered by the attack, and the overhead was therefore not measured.

*Further modifications.* Since the acquisition of power traces is not the main focus of the paper, the software was further modified to make the experiment easier. Particularly, the software allows an iteration-by-iteration execution of the three point ladder which toggles a GPIO pin at the beginning of the `swap_points`

[4] The mentioned port can be found in the official datasheet for the ChipWhisperer's "UFO" board: `http://media.newae.com/datasheets/NAE-CW308-datasheet.pdf`.

function. When switched on, the GPIO notifies the oscilloscope to start the collection of power measurements.

Though these modifications create an unrealistic attack scenario, the experiment is *still* practical on unmodified software but requires additional effort of marginal complexity. In a real-world scenario, the adversary first requires to observe the power consumption of the target device by measuring the current through a shunt resistor in series between the microcontroller and the ground (or the voltage collector). The collection of power samples can then be synchronized on communication which requires an oscilloscope with a buffer of a few hundred million samples to capture the consumption of the entire three point ladder. Finally, the parts which correspond to `swap_points` need to be identified in the collected trace, and then carefully segmented. A reader interested in such a process is advised to read [19].

*Source code.* The final software on which power traces were acquired can be found here: `https://github.com/AymericGenet/SIKE-clusterswap-2021`.

### 5.2 Traces collection

To collect power traces corresponding to `swap_points` executions, a simulation of an ephemeral SIKE key exchange was conducted:

(1) Program the STM32F3 with a random key and seed.
(2) Generate and send three valid points $Q$, $P$, and $Q - P$.
(3) Repeat the following $n$ times:
    (a) Make the STM32F3 execute the next loop iteration.
    (b) Save the power trace from the oscilloscope.

The above was repeated 1,000 times (each time with a different key and seed) for SIKEp434 (hence $n = 218$). An example of a power trace along with its frequency components for SIKEp434 is shown in Fig 3. Note that most of the frequency components are zero due to the limiting analog bandwidth of the oscilloscope (20 MHz).

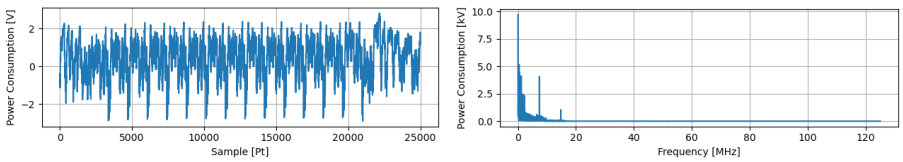



Fig. 3: Example of one of the $n$ power traces corresponding to `swap_points` in a single iteration of the loop (left) along with its Fourier representation (right).

### 5.3 Clustering power analysis

In the next step of the experiment, the $n$ collected traces of each experiment are exploited to attempt a key recovery as explained in Section 3 (cf. Algorithm 2).

(1) Process (for $0 \leq i < n$):
   (a) $T_i$ with an $\ell$-level *wavelet transform* ($\hat{T}_i$ of length $\hat{M} = M/2^\ell$),
   (b) $\hat{T}_i$ with a *Fourier transform* ($\hat{F}_i$).

(2) Run the attack on both $\hat{T}_i$ and $\hat{F}_i$:
   (a) Go to the next sample location $0 \leq t < \hat{M}$ (resp. $0 \leq f < \hat{M}/2$).
   (b) Run clustering on $\{\hat{T}_i[t]\}_{0 \leq i < n}$ (resp. on $\{\hat{F}_i[f]\}_{0 \leq i < n}$).
   (c) Record the $sk_t$ returned for time $t$ (resp. $sk_f$).

The above was repeated with $0 \leq \ell < 8$ levels of wavelet with a Symlet wavelet of filter length 8 (i.e., `sym4`) to further show the efficiency of the processing transforms. The success rate is calculated through all the timing positions and frequencies over the 1,000 sets of measured traces by comparing the recovered key with the correct key.

### 5.4 Results

Out of the 1,000 experiments, across all the levels $0 \leq \ell < 8$, the correct key is *always* found in the set of recovered keys $sk_t$ or $sk_f$. Table 1 and Table 2 report various metrics about how often the correct key appears in the two sets of recovered keys. The independent success rates of each timing position and frequency are reported in Fig. 4. Finally, examples of samples distribution successfully clustered is shown in Fig. 5 both in timing and frequency.

Table 1: Statistics on the total number of timing locations which yield the correct key across the $N = 1{,}000$ experiments.

| | $k$-means | | | | Thresholding | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $\ell$ | min. | max. | $\mathbf{E}(\#t)$ | SD($\#t$) | min. | max. | $\mathbf{E}(\#t)$ | SD($\#t$) | $\hat{M}$ |
| 0 | 154 | 341 | 251.704 | 30.056 | 115 | 289 | 196.668 | 29.366 | 25000 |
| 1 | 72 | 172 | 125.611 | 15.153 | 58 | 144 | 97.923 | 14.764 | 12503 |
| 2 | 37 | 85 | 62.329 | 7.646 | 28 | 71 | 48.469 | 7.582 | 6255 |
| 3 | 15 | 44 | 29.425 | 4.171 | 11 | 36 | 22.958 | 3.971 | 3131 |
| 4 | 8 | 27 | 15.494 | 2.723 | 5 | 21 | 12.127 | 2.502 | 1569 |
| 5 | 6 | 21 | 13.445 | 2.371 | 4 | 18 | 10.531 | 2.195 | 788 |
| 6 | 2 | 12 | 6.036 | 1.615 | 1 | 9 | 4.033 | 1.408 | 397 |
| 7 | 0 | 5 | 1.645 | 0.941 | 0 | 5 | 0.853 | 0.878 | 202 |

Table 2: Statistics on the total number of frequencies which yield the correct key across the $N = 1{,}000$ experiments.

| | $k$-means | | | | Thresholding | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $\ell$ | min. | max. | $\mathbf{E}(\#f)$ | SD$(\#f)$ | min. | max. | $\mathbf{E}(\#f)$ | SD$(\#f)$ | $\hat{M}/2$ |
| 0 | 18 | 29 | 23.625 | 1.774 | 17 | 28 | 21.965 | 1.659 | 12500 |
| 1 | 16 | 26 | 20.704 | 1.630 | 15 | 24 | 19.382 | 1.584 | 6251 |
| 2 | 18 | 27 | 21.900 | 1.460 | 16 | 27 | 20.901 | 1.515 | 3127 |
| 3 | 15 | 30 | 22.641 | 2.288 | 13 | 26 | 19.993 | 2.239 | 1565 |
| 4 | 11 | 20 | 15.001 | 1.429 | 9 | 18 | 13.815 | 1.488 | 784 |
| 5 | 6 | 11 | 8.611 | 0.908 | 5 | 10 | 8.063 | 0.895 | 394 |
| 6 | 3 | 7 | 4.467 | 0.791 | 2 | 7 | 4.162 | 0.749 | 198 |
| 7 | 2 | 6 | 4.023 | 0.800 | 2 | 7 | 3.777 | 0.747 | 101 |

### 5.5 Discussion

The above experiment proves that the recommended Cortex-M4 implementation of SIKE from [47] is vulnerable to low-effort power analyses, even in the case when the implementation is protected with coordinate randomization. As a result, the main objective of the experiment is achieved.

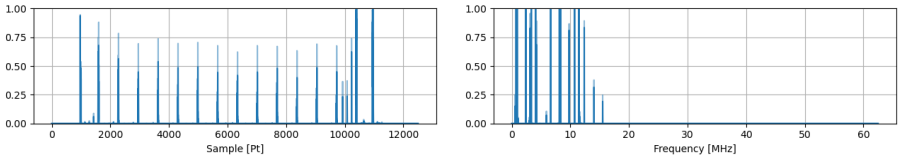
The rest of the discussion focuses on the efficiency of the improvements.

**Wavelet efficiency.** Contrary to expectations, processing the power traces with the wavelet transform does not improve the success rate (cf. Fig. 4). While the wavelet transform features noise filtering, information is still lost during the operation as the convolution involved in the transform combines significant power samples with insignificant ones. Nevertheless, the quality of the compression is fitting as the correct key still occurs on average more than once throughout the timing locations, even after several levels of filtering. Therefore, the number of samples to visit can be reduced by a significant factor using this transform.

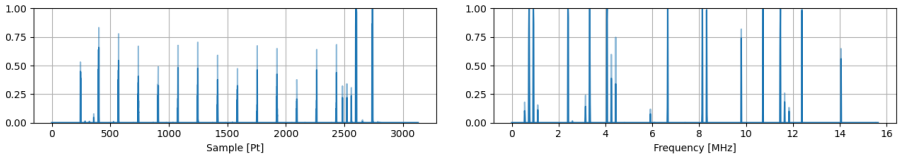
As the power trace corresponds to the execution of the `swap_points` function, the sample locations reported in Fig. 4 correspond to specific instructions of the attacked implementation (shown in Appendix A, Listing 1.3) In particular, the very first spike in the figure correspond to the mask computation which expands a secret bit. The regular spikes in the middle correspond to the swap formula performed on each of the 32-bit words of the points. Finally, the last two spikes at the end of the graph correspond to exiting the function. These spikes show all the aspects of the implementation that need protection.

Note that the choice of the wavelet function (`sym4`) was guided by an experimental exploration and that similar results are expected by using a different family or a different filter length.

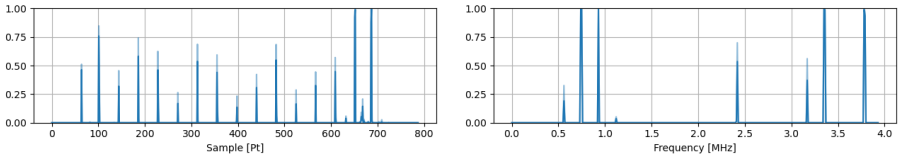
**Fourier efficiency.** The Fourier analysis shows remarkable efficiency across all levels of wavelet transforms. Performing a clustering power analysis with frequency components rather than power samples is shown to have a resounding



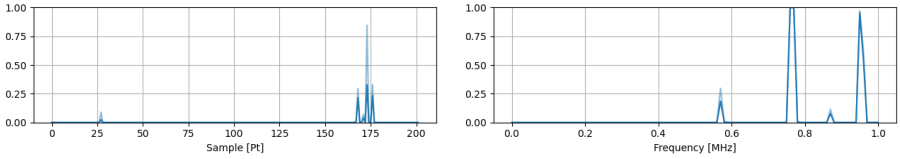

(a) $\ell = 1$




(b) $\ell = 3$




(c) $\ell = 5$




(d) $\ell = 7$

Fig. 4: Success rate of the clustering power analysis (thresholding in opaque vs. $k$-means in transparent) at each timing locations (left) and frequencies (right) across different levels of wavelet transforms.

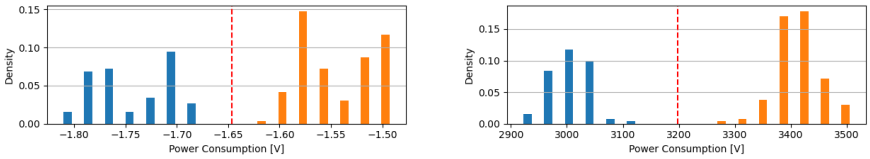
success rate across all experiments. Furthermore, such a success rate is kept throughout the wavelet levels, as the leakage happens at low frequencies which are preserved by the wavelet transform. The most notable observation to make is that clustering in the frequency domain is successful even at the last wavelet level where the same analysis in timing is shown to be inefficient. This proves that even though clustering power samples independently happens to be ineffective, their combination in the frequency domain may be sufficient to perform a successful analysis.

There may be many reasons why low frequencies leak most of the information in the current case. The frequencies of interest are suspected to be subharmonics of the clock speed. For instance, the spikes at 0.92 MHz likely correspond to the pattern of eight instructions in the `swap_points` function (see Listing 1.3). The same can be said for the spikes at 0.73 MHz and 0.74 MHz, as such frequencies

also happen to be a tenth of the clock speed. These frequencies, as well as the other significant ones, may also be due to the consumption of sub-systems in the hardware (e.g., memory) that function at different paces.

**Thresholding efficiency.** In addition to demonstrating the efficiency of the pre-processing phase, the experiments show that the thresholding proposed in Algorithm 3 is almost as successful as $k$-means. While $k$-means still obtains better results (cf. Fig. 4), our experiment with $k$-means took 29 hours to be performed, while the same analysis with thresholding took only 6.5 hours.

**Majority rule efficiency.** The extremely high occurrence of the correct key in Table 1 and Table 2 confirms that the vertical majority rule explained in Section 4 helps validating the key. In all experiments, the most recorded candidate was always observed to be the correct key. As a result, the correct key is expected to be recovered within the first try-outs as the other candidates were all observed to be either random or close to the correct key.




(a) $t = 1943$ (b) $f = 0.92$ [MHz]

Fig. 5: Example of a power sample distributions ($\ell = 0$). The threshold (in red) was found by Algorithm 3.

### 5.6 Other SIKE instances

Note that the success of the clustering is closely connected to the relatively big number of samples available. As more samples are obtained, the distinction between the two clusters becomes easier. However, depending on the noise, additional samples may undermine the success of the overall clustering.

Still, similar results (if not better) have been obtained by running the same experiment with the bigger instances of SIKE. The experiments were executed with fewer runs, a fixed wavelet level, and only using the thresholding algorithm. The results are reported in Table 3 and prove that the attack is not limited to SIKEp434.

Table 3: Statistics on the total number of timing locations and total number of frequencies which yield the correct key across the $N = 10$ experiments with the other instances of SIKE ($\ell = 5$).

| | | Timing | | | | Frequency | | | |
|---|---|---|---|---|---|---|---|---|---|
| $p$ | $n$ | min. | max. | $\mathbf{E}(\#t)$ | SD($\#t$) | min. | max. | $\mathbf{E}(\#f)$ | SD($\#f$) |
| 503 | 252 | 8 | 14 | 10.5 | 2.5 | 9 | 11 | 9.7 | 0.7 |
| 610 | 304 | 16 | 31 | 22.1 | 5.3 | 10 | 14 | 11.7 | 1.3 |
| 751 | 378 | 17 | 25 | 22.2 | 2.7 | 9 | 13 | 10.6 | 1.5 |

## 6 Countermeasure

Protecting the point-swapping procedure against clustering power analysis is not obvious, as the attack defeats classical countermeasures of [13] which include coordinate randomization, exponent randomization, point blinding, and even shuffling the for-loop. Moreover, due to the recent study which relies on deep learning [42], even the tiniest bias in the power consumption may lead to a full recovery.

To make the task even more challenging, the target CPU of the Cortex-M4 is known to be hard to protect (see [5]). As the Cortex-M4 appears to leak in the Hamming distance of the pipeline registers (see [14]), the countermeasures need not only to consider the Hamming weight of the processed values, but also the Hamming distance between the values used by two consecutive instructions.

In this section, a countermeasure based on thresholding the swapping mask is suggested.

### 6.1 Description

The proposed countermeasure revises the original swapping procedure from Section 2 in the following sense; instead of computing the value `mask` & $(a \oplus b)$ all at once, the idea is to split this quantity into two shares and add each share separately in a two-stage process (to both $a$ and $b$). Such a procedure avoids computing values of extreme Hamming distances.

To this end, the swapping mask is replaced by two 32-bit masks: `m1` and `m2` such that their bitwise "xor" is equal to `mask`. In other words, given two consecutive private-key bits $sk_{i-1}$, and $sk_i$ (for $0 \leq i < n$ with $sk_{-1} = sk_n = 0$):

$$\texttt{m1} \oplus \texttt{m2} = \begin{cases} \texttt{0x00000000} & \text{if } sk_{i-1} \oplus sk_i = 0, \\ \texttt{0xFFFFFFFF} & \text{if } sk_{i-1} \oplus sk_i = 1. \end{cases}$$

Given the two masks `m1` and `m2`, the new procedure works as follows:

$$\begin{aligned} \texttt{tmp1} &= \texttt{m1} \ \& \ (a \oplus b), \\ \texttt{tmp2} &= \texttt{m2} \ \& \ (a \oplus b), \\ a &= (\texttt{tmp1} \oplus a) \oplus \texttt{tmp2}, \\ b &= (\texttt{tmp1} \oplus b) \oplus \texttt{tmp2}. \end{aligned}$$

Because of the property of $\texttt{m1} \oplus \texttt{m2}$, the above procedure swaps $a$ and $b$ in the same sense as the method in Section 2.

### 6.2 Implementation

The results from Section 5 provide insight on the critical points of the procedure that require particular care. Mainly three leaking points were identified:

1. The generation of the masks.
2. The instructions used to perform the swapping operation.
3. Exiting the function.

The third point can be avoided by incorporating the procedure to the code without calling a function, so only the first two points are addressed.

**Masks generation.** Let `swap` refer to the secret difference of private-key bits (i.e., $\texttt{swap} = sk_{i-1} \oplus sk_i$). The suggested countermeasure involves generating two random masks `m1` and `m2` that are either equal or bit-wise complement depending on `swap`. To achieve this, given a random `m1`, the second mask `m2` is derived with the following formula: $\texttt{m2} = (1 - 2 \cdot \texttt{swap})(\texttt{m1} + \texttt{swap})$. This makes `m2` become the bitwise complement of `m1` through the representation of negative numbers in the CPU with the two's complement (i.e., $\texttt{m2} = -(\texttt{m1} + 1)$ if $\texttt{swap} = 1$).

*Performance.* Safely generating these two quantities requires sampling additional randomness. In particular, the multiplication of $(\texttt{m1}+\texttt{swap})$ by $(1-2\cdot\texttt{swap})$ is computed as $u_1(\texttt{m1}+\texttt{swap})-u_2(\texttt{m1}+\texttt{swap})$ where $u_1-u_2 = 1-2\cdot\texttt{swap}$. In total the mask generation requires at least 8 bytes of entropy (29 bits of which are effective) and introduces an overhead of at least 12 additional instructions when compared to the original mask computation. The code is given in Listing 1.4.

**The swapping operation.** Because of the Cortex-M4 leakage model, the order of the operations and of the operands play a critical role in the countermeasure. Particular care has to be taken with store and load instructions, as the power consumption of these procedures leaks sensitive values. As a result, given the two masks `m1` and `m2` generated as before, the implementation of the countermeasure must follow a special order given in Listing 1.5.

*Performance.* As opposed to the original pattern of 8 instructions, such a solution requires 14 instructions per iteration and doubles the numbers of loads and stores which introduces further delay.

**Benchmarks.** We compare the runtimes of our countermeasure against the runtimes of the unprotected version of SIKE. About 62% of the overhead stems from acquiring randomness for mask generation. As we generate a new mask for each swap (so for each word), we use a cheap pseudorandom number generator to limit the impact on performance; namely, the Tiny Mersenne Twister pseudorandom number generator [45] seeded with a 64-bit value obtained from a source of true randomness.

The protected `swap_points` is about 5.7 times slower than the unprotected swap. Within the three point ladder function, the overhead adds up to about 70,000 additional cycles which takes up to 5% of the total computing time of the three point ladder. When considered as a part of a full execution of SIKE, the overhead due to protecting the swap boils down from about 1% in the key generation and decapsulation to 0.7% in the encapsulation procedure.

Table 4: Runtimes (in cycles) of the SIKEp434 implementation with and without the countermeasure on an Intel i9-8950HK CPU @ 2.90GHz with Turbo boost turned off.

| Operation | unprotected | protected |
|---|---|---|
| Mask generation | 1 | 251 |
| Swapping operation | 71 | 148 |
| Three point ladder | 1,172,432 | 1,241,721 |
| Key generation | 6,083,645 | 6,153,241 |
| Encapsulation | 9,893,673 | 9,962,113 |
| Decapsulation | 10,625,881 | 10,747,176 |

### 6.3 Experimental validation

The proposed countermeasure was validated by conducting Welch's $t$-test [46]. Such a test gives a degree of confidence that two classes of power samples are statistically indistinguishable. In the present case, the two classes respectively correspond to whether the points were swapped during the collection of the power traces, or not.

The $t$ values are computed with the following formula:

$$t = \frac{\mu_0 - \mu_1}{\sqrt{\sigma_0^2/n_0 + \sigma_1^2/n_1}}$$

where $\mu_0$, $\mu_1$ correspond to the means of the two classes, $\sigma_0^2$, $\sigma_1^2$ to their variances, and $n_0$, $n_1$ to their cardinality (here, $n_0, n_1 \approx 1000$). A threshold of 4.5 for the $t$ values is set to reject the null hypothesis (see [18]). In other words, a $t$ value greater than the threshold gives evidence that the two distributions are not indistinguishable.

The results are shown in Fig. 6. Even though significantly large $t$ values appear in the plots, the attack is still unsuccessful when re-run against the countermeasure as the histograms corresponding to the power samples from the two classes overlap with each other at all points in time and frequency. Fig. 7 illustrates this by showing the histograms at the highest peaks of the $t$-test plots from Fig. 6. As one can notice, both histograms exhibit a significant variance, which prevents the attack to fully recover the private key. The histograms at all other points showed a similar overlapping.

While such a discrepancy of distributions prevents a successful clustering with the techniques described in this paper, more sophisticated attacks (such as [43,42]) may still prevail. These attacks may therefore require additional efforts to withstand.

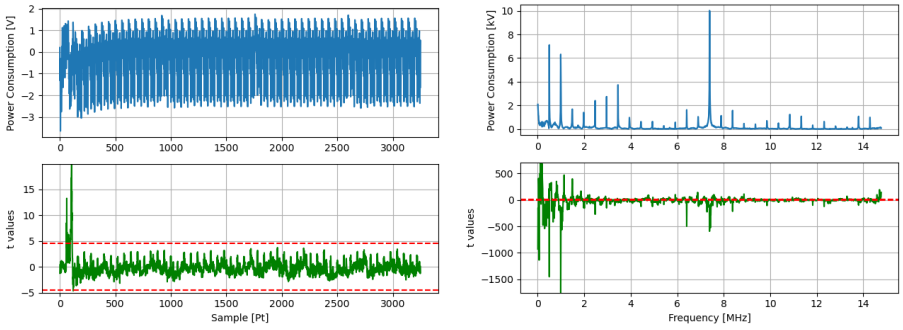
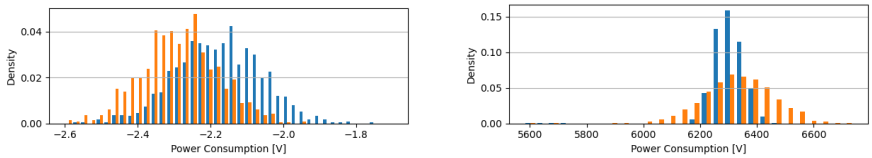



Fig. 6: $t$-test of the countermeasure both in timing and frequency. The horizontal lines in red show the threshold above which the null hypothesis is rejected.




(a) $t = 101$ (b) $f = 0.98$ [MHz]

Fig. 7: Power sample distributions at the locations which produced the highest value in both $t$-tests.

### 6.4 Other countermeasures.

In addition to the countermeasure proposed, other techniques are likely to prevent a clustering power analysis of the swapping procedure. Desynchronizing the clock of the target device results in unaligned power traces with different random frequencies, so the attack is expected to be unsuccessful in neither domains. Such a countermeasure might be implemented by interleaving dummy `nop` instructions with the actual instructions of the regular `swap_points` function. Alternatively, swapping pointer addresses rather than values may be effective in the power domain but is shown to succumb to the same attack using electromagnetic radiations in [37].

## 7 Conclusion & Future work

The paper described a plain clustering power analysis able to recover the entire private key in a single execution of the three point ladder in the implementation of SIKE for Cortex-M4. In particular, the paper demonstrated that processing the traces with a wavelet transform efficiently reduces the number of timing locations to visit, and that clustering frequency components may succeed even where clustering power samples is inefficient.

While the attack has been experimentally shown to be always successful, the reader must keep in mind that the experiment was performed using the ChipWhisperer framework on a chip that was deliberately made vulnerable to power analysis. However, the countermeasure described completely thwarts the attack even on such a vulnerable chip. If the countermeasure is safe under such defenseless circumstances, then the implementation can be assumed to be safe in a more realistic scenario.

As future work, the experiment could be repeated with a different clock speed to evaluate the evolution of the frequency components. Other improvements using, e.g., multiple samples of a single iteration in a multivariate clustering analysis may also be investigated. The proposed countermeasure requires to be evaluated against other side-channel attacks and improved both in performance and security. Finally, an evaluation of other sensitive operations in SIKE—such as the isogeny computation—can be conducted, as there are still many other points that have not been evaluated yet that may also leak secret information through power consumption.

## Acknowledgements

We would like to thank Arjen K. Lenstra and Thorsten Kleinjung for the weekly discussions held on this paper. We are also grateful to Nicolas Oberli, Sylvain Pelissier, Hervé Pelletier, Kopiga Rasiah, Mathilde Raynal, and Karine Villegas for their precious help in the realization of this project. Finally, we want to express our gratitude to the reviewers of COSADE 2022 for their valuable comments.

## References

1. Agrawal, D., Archambeault, B., Rao, J.R., Rohatgi, P.: The EM side-channel(s). In: Jr., B.S.K., Koç, Ç.K., Paar, C. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers. Lecture Notes in Computer Science, vol. 2523, pp. 29–45. Springer (2002), https://doi.org/10.1007/3-540-36400-5_4
2. Alam, M., Yilmaz, B., Werner, F., Samwel, N., Zajic, A., Genkin, D., Yarom, Y., Prvulovic, M.: Nonce@Once: A single-trace EM side channel attack on several constant-time elliptic curve implementations in mobile platforms. In: 6th IEEE European Symposium on Security and Privacy, EuroS&P 2021, September 6-10, 2021. IEEE (2021), https://cs.adelaide.edu.au/~yval/pdfs/AlamYWSZGYP21.pdf
3. Aldaya, A.C., Brumley, B.B.: Online template attacks: Revisited. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2021**(3), 28–59 (2021), https://doi.org/10.46586/tches.v2021.i3.28-59
4. Azouaoui, M., Durvaux, F., Poussier, R., Standaert, F.X., Papagiannopoulos, K., Verneuil, V.: On the worst-case side-channel security of ECC point randomization in embedded devices. In: Bhargavan, K., Oswald, E., Prabhakaran, M.

(eds.) Progress in Cryptology - INDOCRYPT 2020 - 21st International Conference on Cryptology in India, Bangalore, India, December 13-16, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12578, pp. 205–227. Springer (2020), https://doi.org/10.1007/978-3-030-65277-7_9
5. Batina, L., Chmielewski, Ł., Haase, B., Samwel, N., Schwabe, P.: SCA-secure ECC in software - mission impossible? IACR Cryptol. ePrint Arch. p. 1003 (2021), https://eprint.iacr.org/2021/1003
6. Batina, L., Chmielewski, Ł., Papachristodoulou, L., Schwabe, P., Tunstall, M.: Online template attacks. In: Meier, W., Mukhopadhyay, D. (eds.) Progress in Cryptology - INDOCRYPT 2014 - 15th International Conference on Cryptology in India, New Delhi, India, December 14-17, 2014, Proceedings. Lecture Notes in Computer Science, vol. 8885, pp. 21–36. Springer (2014), https://doi.org/10.1007/978-3-319-13039-2_2
7. Bernstein, D.J.: The ChaCha family of stream ciphers (2008), https://cr.yp.to/chacha.html
8. Bohy, L., Neve, M., Samyde, D., Quisquater, J.: Principal and independent component analysis for crypto-systems with hardware unmasked units. In: Proceedings of e-Smart 2003 (2003)
9. Brier, E., Clavier, C., Olivier, F.: Correlation power analysis with a leakage model. In: Joye, M., Quisquater, J.J. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings. Lecture Notes in Computer Science, vol. 3156, pp. 16–29. Springer (2004), https://doi.org/10.1007/978-3-540-28632-5_2
10. Chari, S., Rao, J.R., Rohatgi, P.: Template attacks. In: Jr., B.S.K., Koç, Ç.K., Paar, C. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers. Lecture Notes in Computer Science, vol. 2523, pp. 13–28. Springer (2002), https://doi.org/10.1007/3-540-36400-5_3
11. Charvet, X., Pelletier, H.: Improving the DPA attack using wavelet transform. In: NIST Physical Security Testing Workshop. vol. 46 (2005)
12. Clavier, C., Feix, B., Gagnerot, G., Roussellet, M., Verneuil, V.: Horizontal correlation analysis on exponentiation. In: Soriano, M., Qing, S., López, J. (eds.) Information and Communications Security - 12th International Conference, ICICS 2010, Barcelona, Spain, December 15-17, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6476, pp. 46–61. Springer (2010), https://doi.org/10.1007/978-3-642-17650-0_5
13. Coron, J.S.: Resistance against differential power analysis for elliptic curve cryptosystems. In: Koç, Ç.K., Paar, C. (eds.) Cryptographic Hardware and Embedded Systems, First International Workshop, CHES'99, Worcester, MA, USA, August 12-13, 1999, Proceedings. Lecture Notes in Computer Science, vol. 1717, pp. 292–302. Springer (1999), https://doi.org/10.1007/3-540-48059-5_25
14. Corre, Y.L., Großschädl, J., Dinu, D.: Micro-architectural power simulator for leakage assessment of cryptographic software on ARM Cortex-M3 processors. In: Fan, J., Gierlichs, B. (eds.) Constructive Side-Channel Analysis and Secure Design - 9th International Workshop, COSADE 2018, Singapore, April 23-24, 2018, Proceedings. Lecture Notes in Computer Science, vol. 10815, pp. 82–98. Springer (2018), https://doi.org/10.1007/978-3-319-89641-0_5
15. Costello, C., Smith, B.: Montgomery curves and their arithmetic - the case of large characteristic fields. J. Cryptogr. Eng. **8**(3), 227–240 (2018), https://doi.org/10.1007/s13389-017-0157-6

16. Danger, J.L., Guilley, S., Hoogvorst, P., Murdica, C., Naccache, D.: Improving the Big Mac attack on elliptic curve cryptography. In: Ryan, P.Y.A., Naccache, D., Quisquater, J.J. (eds.) The New Codebreakers - Essays Dedicated to David Kahn on the Occasion of His 85th Birthday. Lecture Notes in Computer Science, vol. 9100, pp. 374–386. Springer (2016), https://doi.org/10.1007/978-3-662-49301-4_23
17. De Mulder, E., Buysschaert, P., Ors, S., Delmotte, P., Preneel, B., Vandenbosch, G., Verbauwhede, I.: Electromagnetic analysis attack on an FPGA implementation of an elliptic curve cryptosystem. In: EUROCON 2005 - The International Conference on "Computer as a Tool". vol. 2, pp. 1879–1882 (2005). https://doi.org/10.1109/EURCON.2005.1630348
18. Ding, A.A., Zhang, L., Durvaux, F., Standaert, F., Fei, Y.: Towards sound and optimal leakage detection procedure. In: Eisenbarth, T., Teglia, Y. (eds.) Smart Card Research and Advanced Applications - 16th International Conference, CARDIS 2017, Lugano, Switzerland, November 13-15, 2017, Revised Selected Papers. Lecture Notes in Computer Science, vol. 10728, pp. 105–122. Springer (2017). https://doi.org/10.1007/978-3-319-75208-2_7, https://doi.org/10.1007/978-3-319-75208-2_7
19. Dugardin, M., Papachristodoulou, L., Najm, Z., Batina, L., Danger, J.L., Guilley, S.: Dismantling real-world ECC with horizontal and vertical template attacks. In: Standaert, F.X., Oswald, E. (eds.) Constructive Side-Channel Analysis and Secure Design - 7th International Workshop, COSADE 2016, Graz, Austria, April 14-15, 2016, Revised Selected Papers. Lecture Notes in Computer Science, vol. 9689, pp. 88–108. Springer (2016), https://doi.org/10.1007/978-3-319-43283-0_6
20. Dunn, J.C.: A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. Journal of Cybernetics **3**(3), 32–57 (1973). https://doi.org/10.1080/01969727308546046, https://doi.org/10.1080/01969727308546046
21. Faz-Hernández, A., López-Hernández, J.C., Ochoa-Jiménez, E., Rodríguez-Henríquez, F.: A faster software implementation of the supersingular isogeny diffie-hellman key exchange protocol. IEEE Trans. Computers **67**(11), 1622–1636 (2018), https://doi.org/10.1109/TC.2017.2771535
22. Feo, L.D., Jao, D., Plût, J.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. J. Math. Cryptol. **8**(3), 209–247 (2014), https://doi.org/10.1515/jmc-2012-0015
23. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. J. Cryptol. **26**(1), 80–101 (2013), https://doi.org/10.1007/s00145-011-9114-1
24. Gélin, A., Wesolowski, B.: Loop-abort faults on supersingular isogeny cryptosystems. In: Lange, T., Takagi, T. (eds.) Post-Quantum Cryptography - 8th International Workshop, PQCrypto 2017, Utrecht, The Netherlands, June 26-28, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10346, pp. 93–106. Springer (2017), https://doi.org/10.1007/978-3-319-59879-6_6
25. Genêt, A., de Guertechin, N.L., Kaluderovic, N.: Full key recovery side-channel attack against ephemeral SIKE on the Cortex-M4. In: Bhasin, S., Santis, F.D. (eds.) Constructive Side-Channel Analysis and Secure Design - 12th International Workshop, COSADE 2021, Lugano, Switzerland, October 25-27, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12910, pp. 228–254. Springer (2021), https://doi.org/10.1007/978-3-030-89915-8_11

26. Heuser, A., Zohner, M.: Intelligent machine homicide - breaking cryptographic devices using support vector machines. In: Schindler, W., Huss, S.A. (eds.) Constructive Side-Channel Analysis and Secure Design - Third International Workshop, COSADE 2012, Darmstadt, Germany, May 3-4, 2012. Proceedings. Lecture Notes in Computer Science, vol. 7275, pp. 249–264. Springer (2012), https://doi.org/10.1007/978-3-642-29912-4_18
27. Heyszl, J., Ibing, A., Mangard, S., Santis, F.D., Sigl, G.: Clustering algorithms for non-profiled single-execution attacks on exponentiations. In: Francillon, A., Rohatgi, P. (eds.) Smart Card Research and Advanced Applications - 12th International Conference, CARDIS 2013, Berlin, Germany, November 27-29, 2013. Revised Selected Papers. Lecture Notes in Computer Science, vol. 8419, pp. 79–93. Springer (2013), https://doi.org/10.1007/978-3-319-08302-5_6
28. Heyszl, J., Mangard, S., Heinz, B., Stumpf, F., Sigl, G.: Localized electromagnetic analysis of cryptographic implementations. In: Dunkelman, O. (ed.) Topics in Cryptology - CT-RSA 2012 - The Cryptographers' Track at the RSA Conference 2012, San Francisco, CA, USA, February 27 - March 2, 2012. Proceedings. Lecture Notes in Computer Science, vol. 7178, pp. 231–244. Springer (2012). https://doi.org/10.1007/978-3-642-27954-6_15, https://doi.org/10.1007/978-3-642-27954-6_15
29. Jao, D., Azarderakhsh, R., Campagna, M., Costello, C., Feo, L.D., Hess, B., Hutchinson, A., Jalali, A., Karabina, K., Koziel, B., LaMacchia, B., Longa, P., Naehrig, M., Pereira, G., Renes, J., Soukharev, V., Urbanik, D.: SIKE – supersingular isogeny key exchange (2017), https://sike.org/
30. Kocher, P.C., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M.J. (ed.) Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings. Lecture Notes in Computer Science, vol. 1666, pp. 388–397. Springer (1999), https://doi.org/10.1007/3-540-48405-1_25
31. Kocher, P.C., Jaffe, J., Jun, B., Rohatgi, P.: Introduction to differential power analysis. J. Cryptogr. Eng. **1**(1), 5–27 (2011), https://doi.org/10.1007/s13389-011-0006-y
32. Koziel, B., Azarderakhsh, R., Jao, D.: Side-channel attacks on quantum-resistant supersingular isogeny Diffie–Hellman. In: SAC (2017)
33. Lerman, L., Bontempi, G., Markowitch, O.: Power analysis attack: an approach based on machine learning. Int. J. Appl. Cryptogr. **3**(2), 97–115 (2014), https://doi.org/10.1504/IJACT.2014.062722
34. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability. vol. 1, pp. 281–297. Oakland, CA, USA (1967)
35. Mallat, S.: A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way. Academic Press, Inc., USA, 3rd edn. (2008)
36. Medwed, M., Oswald, E.: Template attacks on ECDSA. In: Chung, K.I., Sohn, K., Yung, M. (eds.) Information Security Applications, 9th International Workshop, WISA 2008, Jeju Island, Korea, September 23-25, 2008, Revised Selected Papers. Lecture Notes in Computer Science, vol. 5379, pp. 14–27. Springer (2008), https://doi.org/10.1007/978-3-642-00306-6_2
37. Nascimento, E., Chmielewski, Ł.: Applying horizontal clustering side-channel attacks on embedded ECC implementations. In: Eisenbarth, T., Teglia, Y. (eds.) Smart Card Research and Advanced Applications - 16th International Conference, CARDIS 2017, Lugano, Switzerland, November 13-15, 2017, Revised Selected Pa-

pers. Lecture Notes in Computer Science, vol. 10728, pp. 213–231. Springer (2017), https://doi.org/10.1007/978-3-319-75208-2_13
38. Nascimento, E., Chmielewski, Ł., Oswald, D.F., Schwabe, P.: Attacking embedded ECC implementations through cmov side channels. In: Avanzi, R., Heys, H.M. (eds.) Selected Areas in Cryptography - SAC 2016 - 23rd International Conference, St. John's, NL, Canada, August 10-12, 2016, Revised Selected Papers. Lecture Notes in Computer Science, vol. 10532, pp. 99–119. Springer (2016), https://doi.org/10.1007/978-3-319-69453-5_6
39. NIST Computer Security Division: Post-quantum cryptography standardization (2016), https://csrc.nist.gov/Projects/Post-Quantum-Cryptography
40. NIST Computer Security Division: Post-quantum cryptography standardization – round 3 submissions (2021), https://csrc.nist.gov/Projects/post-quantum-cryptography/round-3-submissions
41. Perin, G., Chmielewski, Ł.: A semi-parametric approach for side-channel attacks on protected RSA implementations. In: Homma, N., Medwed, M. (eds.) Smart Card Research and Advanced Applications - 14th International Conference, CARDIS 2015, Bochum, Germany, November 4-6, 2015. Revised Selected Papers. Lecture Notes in Computer Science, vol. 9514, pp. 34–53. Springer (2015), https://doi.org/10.1007/978-3-319-31271-2_3
42. Perin, G., Chmielewski, Ł., Batina, L., Picek, S.: Keep it unsupervised: Horizontal attacks meet deep learning. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2021**(1), 343–372 (2021), https://doi.org/10.46586/tches.v2021.i1.343-372
43. Perin, G., Imbert, L., Torres, L., Maurine, P.: Attacking randomized exponentiations using unsupervised learning. In: Prouff, E. (ed.) Constructive Side-Channel Analysis and Secure Design - 5th International Workshop, COSADE 2014, Paris, France, April 13-15, 2014. Revised Selected Papers. Lecture Notes in Computer Science, vol. 8622, pp. 144–160. Springer (2014), https://doi.org/10.1007/978-3-319-10175-0_11
44. Poussier, R., Zhou, Y., Standaert, F.X.: A systematic approach to the side-channel analysis of ECC implementations with worst-case horizontal attacks. In: Fischer, W., Homma, N. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10529, pp. 534–554. Springer (2017), https://doi.org/10.1007/978-3-319-66787-4_26
45. Saito, M., Matsumoto, M.: Tiny Mersenne Twister pseudo-random number generator (2011), https://github.com/MersenneTwister-Lab/TinyMT
46. Schneider, T., Moradi, A.: Leakage assessment methodology - extended version. J. Cryptogr. Eng. **6**(2), 85–99 (2016), https://doi.org/10.1007/s13389-016-0120-y
47. Seo, H., Anastasova, M., Jalali, A., Azarderakhsh, R.: Supersingular isogeny key encapsulation (SIKE) round 2 on ARM Cortex-M4. Cryptology ePrint Archive, Report 2020/410 (2020), https://eprint.iacr.org/2020/410
48. Shi, F., Wei, J., Sun, D., Wei, G.: A systematic approach to horizontal clustering analysis on embedded RSA implementation. In: 25th IEEE International Conference on Parallel and Distributed Systems, ICPADS 2019, Tianjin, China, December 4-6, 2019. pp. 901–906. IEEE (2019), https://doi.org/10.1109/ICPADS47876.2019.00132
49. Shor, P.W.: 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. pp. 124–134. IEEE Computer Society (1994), https://doi.org/10.1109/SFCS.1994.365700

50. Sim, B.Y., Han, D.G.: Key bit-dependent attack on protected PKC using a single trace. In: Liu, J.K., Samarati, P. (eds.) Information Security Practice and Experience - 13th International Conference, ISPEC 2017, Melbourne, VIC, Australia, December 13-15, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10701, pp. 168–185. Springer (2017), https://doi.org/10.1007/978-3-319-72359-4_10
51. Souissi, Y., Aabid, M.A.E., Debande, N., Guilley, S., Danger, J.L.: Novel applications of wavelet transforms based side-channel analysis. In: Non-Invasive Attack Testing Workshop (11 2021)
52. Specht, R., Heyszl, J., Kleinsteuber, M., Sigl, G.: Improving non-profiled attacks on exponentiations based on clustering and extracting leakage from multi-channel high-resolution EM measurements. In: Mangard, S., Poschmann, A.Y. (eds.) Constructive Side-Channel Analysis and Secure Design - 6th International Workshop, COSADE 2015, Berlin, Germany, April 13-14, 2015. Revised Selected Papers. Lecture Notes in Computer Science, vol. 9064, pp. 3–19. Springer (2015), https://doi.org/10.1007/978-3-319-21476-4_1
53. Tasso, É., Feo, L.D., Mrabet, N.E., Pontié, S.: Resistance of isogeny-based cryptographic implementations to a fault attack. In: Bhasin, S., Santis, F.D. (eds.) Constructive Side-Channel Analysis and Secure Design - 12th International Workshop, COSADE 2021, Lugano, Switzerland, October 25-27, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12910, pp. 255–276. Springer (2021), https://doi.org/10.1007/978-3-030-89915-8_12
54. Ti, Y.B.: Fault attack on supersingular isogeny cryptosystems. In: Lange, T., Takagi, T. (eds.) Post-Quantum Cryptography - 8th International Workshop, PQCrypto 2017, Utrecht, The Netherlands, June 26-28, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10346, pp. 107–122. Springer (2017), https://doi.org/10.1007/978-3-319-59879-6_7
55. Vélu, J.: Isogénies entre courbes elliptiques. Comptes-Rendus de l'Académie des Sciences, Série I **273**, 238–241 (7 1971)
56. Veyrat-Charvillon, N., Gérard, B., Standaert, F.X.: Soft analytical side-channel attacks. In: Sarkar, P., Iwata, T. (eds.) 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. Lecture Notes in Computer Science, vol. 8873, pp. 282–296. Springer (2014), https://doi.org/10.1007/978-3-662-45611-8_15
57. Walter, C.D.: Sliding windows succumbs to Big Mac attack. In: Koç, Ç.K., Naccache, D., Paar, C. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2001, Third International Workshop, Paris, France, May 14-16, 2001, Proceedings. Lecture Notes in Computer Science, vol. 2162, pp. 286–299. Springer (2001), https://doi.org/10.1007/3-540-44709-1_24
58. Zhang, F., Yang, B., Dong, X., Guilley, S., Liu, Z., He, W., Zhang, F., Ren, K.: Side-channel analysis and countermeasure design on ARM-based quantum-resistant SIKE. IEEE Transactions on Computers **69**(11), 1681–1693 (2020), https://doi.org/10.1109/TC.2020.3020407

# A Attacked code

```
for (i = 0; i < nbits; i++) {
  bit = (sk[i>>LOG2RADIX] >> (i&(RADIX-1))) & 1;
  swap = bit ^ prevbit;
  prevbit = bit;
  mask = 0 - (digit_t)swap;

  swap_points(R, R2, mask);
  xDBLADD(R0, R2, R->X, A24);
  randomize_coordinates(R0, R, R2);
}
swap = 0 ^ prevbit;
mask = 0 - (digit_t)swap;
swap_points(R, R2, mask);

return R2;
```

Listing 1.1: Attacked source code of the three point ladder in C (simplified).

```
for (i = 0; i < NWORDS_FIELD; i++) {
  temp = mask & (R->X[0][i] ^ R2->X[0][i]);
  R->X[0][i] = temp ^ R->X[0][i];
  R2->X[0][i] = temp ^ R2->X[0][i];
  temp = mask & (R->Z[0][i] ^ R2->Z[0][i]);
  R->Z[0][i] = temp ^ R->Z[0][i];
  R2->Z[0][i] = temp ^ R2->Z[0][i];
  temp = mask & (R->X[1][i] ^ R2->X[1][i]);
  R->X[1][i] = temp ^ R->X[1][i];
  R2->X[1][i] = temp ^ R2->X[1][i];
  temp = mask & (R->Z[1][i] ^ R2->Z[1][i]);
  R->Z[1][i] = temp ^ R->Z[1][i];
  R2->Z[1][i] = temp ^ R2->Z[1][i];
}
```

Listing 1.2: Attacked source code of `swap_points` in C (simplified).

```
rsb   r8, r6, #0    /* mask = (0 - swap) */
add.w r2, r4, #92   /* R->X  */
add.w r3, r4, #540  /* R2->X */
mov.w ip, #0        /* i = 0 */
<loop>:
ldr   r7, [r2, #0]
ldr   r1, [r3, #0]
eor.w r0, r7, r1    /* mask & (R->X[0][i] ^ R2->X[0][i]) */
and.w r0, r0, r8
eors  r7, r0        /* R->X[0][i] = temp ^ R->X[0][i] */
eors  r1, r0        /* R2->X[0][i] = temp ^ R2->X[0][i] */
str.w r7, [r2], #4
str.w r1, [r3], #4

...                 /* repeat above (with different offsets) */

add.w ip, ip, #1    /* i++ */
cmp.w ip, #14       /* i < NWORDS_FIELD */
bne.n <loop>
```

Listing 1.3: Compiled `swap_points` function with annotations (SIKEp434).

```
and.w %[u1], %[u1], #0xFFFFFFFD  /* u1 = randombytes(4) & 0xFFFFFFFD */
and.w %[m1], %[u2], #0xFFFFFFFE  /* m1 = randombytes(4) & 0xFFFFFFFE */
add.w %[u2], %[u1], %[swap]      /* u2 = u1 + swap */
add.w %[m2], %[m1], %[swap]      /* r  = m1 + swap */
add.w %[u1], %[u1], #1           /* u1 = u1 + 1 */
mul.w %[u1], %[u1], %[m2]        /* u1 = u1*r */
add.w %[u2], %[u2], %[swap]      /* u2 = u2 + swap */
mul.w %[u2], %[u2], %[m2]        /* u2 = u2*r */
sub.w %[m2], %[u1], %[u2]        /* m2 = u1 - u2 */
```

Listing 1.4: Source code of the secure masks generation in assembly.

```
ldr.w %[a], [%[R]]          /* a = R[i]  */
ldr.w %[b], [%[R2]]         /* b = R2[i] */
eor.w %[tmp1], %[a], %[b]   /* tmp1 = a ^ b */
and.w %[tmp1], %[m1]        /* tmp1 = tmp1 & m1 */
eor.w %[b], %[b], %[tmp1]   /* a = a ^ tmp1 */
eor.w %[a], %[a], %[tmp1]   /* b = b ^ tmp1 */
eor.w %[tmp2], %[a], %[b]   /* tmp2 = a ^ b */
str.w %[b], [%[R2]]         /* R2[i] = b */
and.w %[tmp2], %[m2]        /* tmp2 = tmp2 & m2 */
str.w %[a], [%[R]]          /* R[i]  = a */
eor.w %[b], %[b], %[tmp2]   /* b = b ^ tmp2 */
eor.w %[a], %[a], %[tmp2]   /* a = a ^ tmp2 */
str.w %[a], [%[R]], #4      /* R[i]  = a */
str.w %[b], [%[R2]], #4     /* R2[i] = b */

...                         /* repeat above */
```

Listing 1.5: Source code of the secure swapping operation in assembly.