SIKE Channels

Zero-Value Side-Channel Attacks on SIKE

Luca De Feo[1], Nadia El Mrabet[2], Aymeric Genêt[3][4], Novak Kaluđerović[3], Natacha Linard de Guertechin[5], Simon Pontié[6] and Élise Tasso[6]

[1] IBM Research Europe, Zürich, Switzerland, ches22@defeo.lu
[2] Mines Saint-Étienne, CEA-Tech, Centre CMP, F-13541 Gardanne, France, nadia.el-mrabet@emse.fr
[3] École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, aymeric.genet@epfl.ch, novak.kaluderovic@epfl.ch
[4] Nagra Kudelski Group, Cheseaux-sur-Lausanne, Switzerland, aymeric.genet@nagra.com
[5] CYSEC SA, Lausanne, Switzerland, natacha.linard@cysec.com
[6] CEA Tech, Centre CMP, Équipe Commune CEA Tech - Mines Saint-Étienne, F-13541 Gardanne, France
Université Grenoble Alpes, CEA, Leti, F-38000 Grenoble, France, elise.tasso2@cea.fr, simon.pontie@cea.fr

Abstract. We present new side-channel attacks on SIKE, the isogeny-based candidate in the NIST PQC competition. Previous works had shown that SIKE is vulnerable to differential power analysis and pointed to coordinate randomization as an effective countermeasure. We show that coordinate randomization alone is not sufficient, as SIKE is vulnerable to a class of attacks similar to refined power analysis in elliptic curve cryptography, named *zero-value attacks*. We describe and confirm in the lab two such attacks leading to full key recovery, and analyze their countermeasures.

Keywords: SIKE · side-channel attack · zero-value attack · countermeasures · post-quantum cryptography · isogeny-based cryptography

1 Introduction

With the Post-Quantum Cryptography Standardization process nearing the end of its third round, NIST has repeatedly called for side-channel and fault-injection analyses of candidates [Moo18, Apo20]. The great diversity of algorithms opens the way to a variety of attack techniques, which the community is only starting to explore [EFGT17, RHHM17, CMP18, BP18, PSKH18, BDE+18, MGTF19, KL19, SBWE20, GJN20, LNPS20, CCD+21, PP21, SRSWZ21, BDK+21, XPSR+21, UXT+22, XIU+21].

This work is concerned with the side-channel security of SIKE [JAC+20], the lone isogeny-based candidate in the standardization process. SIKE is an IND-CCA2 key encapsulation mechanism (KEM), derived from the key exchange scheme SIDH [JD11] by applying a variant of the Fujisaki–Okamoto transform [FO99, FO13, HHK17]. In SIDH, each party performs a public key generation step followed by a session key derivation step. These steps are all very similar to each other, and are mainly composed of *elliptic curve point operations* and *isogeny evaluations*. The same steps are rearranged in SIKE to form the *key generation*, *encapsulation*, and *decapsulation* algorithms.

Due to its similarity to pre-quantum Elliptic Curve Cryptography (ECC), it is often argued that the same physical attacks and countermeasures that apply to ECC also apply to SIKE. Indeed, timing and simple power analyses are easily countered with constant-time code, which SIKE's reference implementations already provide (setting bugs aside [GJN20]).

Accordingly, full key recoveries through Differential (DPA) and Correlation Power Analysis (CPA) on the elliptic point scalar multiplication part have been demonstrated by several authors [ZYD$^+$20, GLK21]. These attacks rely on the fact that the scalar multiplication in SIKE is a deterministic routine whose inputs, with the only exception of the private key, are publicly known. Thus, by making guesses on independent bits of the private key, it is possible to predict the Hamming weight of the values stored in CPU registers or transferred between memory and registers. This leakage prediction can then be correlated to the power consumption or electromagnetic (EM) emissions to validate bit values.

A cheap and effective countermeasure against DPA and CPA, already commonplace in ECC, is *coordinate randomization* [Cor99]. Elliptic curve points in SIKE are represented non-uniquely by ratios of finite field elements, i.e., the pairs $[X : Z]$ and $[\lambda X : \lambda Z]$ represent the same point for nonzero λ. By randomizing the projective representation of the input points with a random nonzero λ at the beginning of the scalar multiplication, Hamming weights and other observables of computed values are no longer predictable, and thus the correlation with power consumption is lost. Coordinate randomization in SIKE was recommended as a countermeasure in [KPHS18, ZYD$^+$20, GLK21]. Costello [Cos21, §5] even argues that coordinate randomization is likely enough to protect SIKE against most side-channel attacks.

Without fully contradicting Costello, we show that coordinate randomization is not sufficient to protect SIKE against side-channel attacks if additional countermeasures are not implemented. Indeed, attacks bypassing coordinate randomization were already introduced in ECC by Goubin [Gou03], Akishita and Takagi [AT03], and Izu and Takagi [IT03]. As highlighted in [SCDJB21], all these attacks target the emergence of zeros as intermediate values in elliptic curve point computations: because coordinate randomization only randomizes nonzero coordinates, bits of the private key can again be recovered by identifying the moments during which the computation of a zero value occurs. Such a computation is usually recognized in power consumption or EM activity. We shall collectively refer to these as *zero-value attacks*.

Attack scenario. We target a *static key* version of SIKE where a single secret key is used to decrypt several ciphertexts. We assume an implementation that matches SIKE's reference one, and additionally performs coordinate randomization. By appropriately crafting ciphertexts and measuring the power consumption or EM emissions of the decapsulation routine, we are able to recover the majority of the secret key bits, leaving only a few bits of known positions to brute-force. Thus, our attack may be described as a chosen-ciphertext attack with side-channel information.

SIDH and the basic IND-CPA encryption derived from it are well known to be mathematically broken in a chosen ciphertext scenario [GPST16], which is the reason why SIKE relies on the Fujisaki–Okamoto (FO) transform to achieve IND-CCA2 security. The purpose of the transform is precisely to validate that the ciphertext was generated honestly and abort if not. However, the side-channel leakage that we exploit happens *before* FO can prevent the attack. In other words, the cavalry arrives late.

Full validations of SIDH ciphertexts (and public keys) is a problem believed to be as hard as breaking SIDH/SIKE itself [Tho17, GV18, UJ20]. Luckily, our attack can be blocked by a partial form of ciphertext validation, although this countermeasure is not exactly cheap.

Related work. The possibility of applying zero-value attacks to SIKE was already postulated by Koziel, Azarderakhsh, and Jao [KAJ17]. Their work targets a static key version of SIDH with FO transform, however it predates the release of the official SIKE specification and assumes that a form of partial ciphertext validation, such as the one that manages to block our attack, is also performed. They thus concentrate their efforts on devising zero-value attacks that work *despite partial ciphertext validation*, and suggest four such attacks. We shall argue, however, that none of them applies to the current SIKE standard.

The zero-value threat was also hinted at in [GLK21], however the authors similarly conclude that such attacks would be blocked by partial ciphertext validation and do not explore the matter further. Thus, despite a general awareness of the possibility of zero-value attacks prior to this work, no one had ever demonstrated such an attack on SIDH or SIKE.

For completeness, we must also mention fault-injection attacks. Gélin and Wesolowski proposed a loop-abort attack targeting the isogeny evaluation in SIKE [GW17]. By injecting a random value in a loop counter, they make the computation break out of the loop earlier than expected, leading to a bit-by-bit key recovery. The attack was never confirmed in practice but is easily countered by adding checks to ensure that loops are fully executed until the end.

Finally, Ti proposed a key recovery attack based on injecting a random value in a point coordinate [Ti17]. Ti's attack model is peculiar in that it assumes that a static private key is used at least twice to generate the associated public key. It was confirmed experimentally in [TDFEMP21], where a cheap countermeasure is also given.

Our contribution. We introduce and confirm in the lab two zero-value attacks against the official (uncompressed) SIKE implementation hardened with coordinate randomization. Both attacks consist of recovering the private key in an extend-and-prune fashion with special inputs that force intermediate values to be zero depending on a single secret bit. Although the possibility of this threat had been postulated previously, to the best of our knowledge, this is the first complete description of such attacks.

The first attack targets zero values in the elliptic point scalar multiplication part of SIKE, and is thus very close to previously known zero-value attacks on ECC [AT03, SCDJB21]. The second attack targets the isogeny evaluation part of SIKE, which appears to have much greater side-channel leakage, permitting key recovery with as few as one trace per bit. For the second attack, we analyze the behavior of 3- and 4-isogeny evaluation formulas on invalid points, which appears to be novel in itself.

Our attacks are easily, though not so cheaply, countered by partially validating ciphertexts. Koziel, Azarderakhsh, and Jao [KAJ17] proposed zero-value attacks that pass partial validations, however we show these attacks do not apply to standard SIKE. The question of whether other zero-value attacks could bypass ciphertext validation is left open.

We made our code and data available on the following online repository: https://anonymous.4open.science/r/SIKE-SCA-2022-E500.

Plan. We cover the background on Montgomery curves, isogenies, and SIKE in Section 2. Then, Section 3 introduces our attacks in an abstract way, while Section 4 reports on their experimental realizations. Finally, Section 5 discusses the countermeasures, so the paper can close with Section 6.

2 Background on SIKE

SIKE, the Supersingular Isogeny Key Encapsulation, is a post-quantum KEM built on the theory of isogeny graphs of supersingular elliptic curves. We only review here the necessary background to understand zero-value attacks. For a complete tutorial on SIKE, see [Cos19]; for a general overview of isogeny-based cryptography, see [DF17].

2.1 Supersingular curves

All algebraic computations in SIKE take place in a finite field $\mathbb{F}_{p^2} := \mathbb{F}_p[X]/(X^2+1)$, where p is a prime of the form $p = 2^{e_2} \cdot 3^{e_3} - 1$, with $2^{e_2} \approx 3^{e_3} \approx \sqrt{p}$. All elliptic curves used in SIKE can be put in *Montgomery form*

$$E_A \;:\; y^2 = x^3 + Ax^2 + x \qquad\qquad A \in \mathbb{F}_{p^2} \setminus \{\pm 2\}. \tag{1}$$

Such curves are *supersingular* if and only if their group of points has order $(p+1)^2 = 2^{2e_2} \cdot 3^{2e_3}$, in which case the group structure is

$$E_A(\mathbb{F}_{p^2}) \simeq (\mathbb{Z}/2^{e_2}\mathbb{Z})^2 \times (\mathbb{Z}/3^{e_3}\mathbb{Z})^2.$$

Following [CLN16], the curve E_A is represented *projectively* in one of two possible ways, depending on the subroutine:

- As a pair $[A_{24}^+ : C_{24}]$ such that $C_{24} \neq 0$ and $A = (4A_{24}^+ - 2C_{24})/C_{24}$; or
- As a pair $[A_{24}^+ : A_{24}^-]$ such that $A_{24}^+ \neq A_{24}^-$ and $A = 2(A_{24}^+ + A_{24}^-)/(A_{24}^+ - A_{24}^-)$.

Considering invalid internal states caused by malicious ciphertexts, an elliptic curve represented in SIKE may thus fall into one of the following categories:

The *undefined* curve, represented by $[A_{24}^+ : C_{24}] = [A_{24}^+ : A_{24}^-] = [0 : 0]$. This does not represent any algebraic object.

The *degenerate* curve, represented by $C_{24} = 0$ and $A_{24}^+ = A_{24}^- \neq 0$. This is not, properly speaking, a curve.

The *singular* curves with $A = \pm 2$, corresponding to $A_{24}^+ = C_{24} \neq 0$ and $A_{24}^- = 0$, or to $A_{24}^- = C_{24} \neq 0$ and $A_{24}^+ = 0$. These are not elliptic curves, because they exhibit a singularity in $(\mp 1, 0)$ and behave often as exceptional points in formulas.

Elliptic curves, for any value $A \neq \pm 2$. These further subdivide into *ordinary* and *supersingular* curves. Of the $p^2 - 2$ possible values for A, only $\approx p/2$ yield a supersingular curve.

The points of the curve E_A are the projective solutions of Eq. (1). SIKE drops the information on the y-coordinate, and represents them as pairs $[X : Z]$, with X and Z not both null. We shall make a slight abuse of language by calling $[X : Z]$ a point, given that it may correspond to up to two solution of Eq. (1). Considering invalid internal states, an elliptic point in SIKE may be one of

The *undefined* point $[0 : 0]$. This does not correspond to any algebraic point.

The point at infinity $\mathcal{O} = [X : 0]$ with $X \neq 0$, the identity of the elliptic group law.

The *distinguished* point $T = [0 : Z]$, with $Z \neq 0$, of order 2. Assuming the curve is well defined, $[2]T = \mathcal{O}$.

The *special* 4-torsion points $[X : \pm X]$, with $X \neq 0$. Assuming the curve is well defined $[2]P = T$ for any such point.

An ordinary point $[X : Z]$ not belonging to any of the above. Assuming the curve is well defined, such a point is *on the curve* if $X/Z + A + Z/X$ is a square in $\mathbb{F}_{p^2}$. Its algebraic properties, such as its group order, depend on all of X, Z and A.

Whenever P denotes a point, we let X_P and Z_P denote its coordinates, and if $Z_P \neq 0$ we write $x_P := X_P/Z_P$.

It is easy to see that any solution $(x, y) \neq (0, 0)$ to Eq. (1) uniquely determines A. More subtly, given three coordinates x_P, x_Q and x_R, none being zero, there is a unique curve E_A such that E_A contains the points $[x_P : 1]$, $[x_Q : 1]$, $[x_R : 1]$ and the points are co-linear, which implies that $R = Q - P$. This fact is used in SIKE to encode a Montgomery curve E_A as a triple $(P, Q, Q - P)$, with $P, Q, Q - P \notin \{\mathcal{O}, T\}$.

2.2 Scalar multiplication

A large part of SIKE consists in evaluating the group law of some Montgomery curves. Doubling and tripling of a point $[X : Z]$ are given by the formulas

$$[2][X : Z] = \left[(X^2 - Z^2)^2 : 4XZ(X^2 + AXZ + Z^2)\right], \tag{2}$$

$$[3][X : Z] = \left[(X^4 - 6X^2Z^2 - 4AXZ^3 - 3Z^4)^2X : (3X^4 + 4AX^3Z + 6X^2Z^2 - Z^4)^2Z\right]. \tag{3}$$

A key step in SIKE is, given two points P, Q and an integer m, to compute $P + [m]Q$. In practice, because SIKE uses Montgomery $[X : Z]$-coordinates, it encodes the points as the triplet $(P, Q, Q - P)$, and computes the result using a constant-time *three-point ladder* introduced in [FHLOJRH18], described in Algorithm 1.

Algorithm 1: Three-point ladder

```
Input: A Montgomery curve $E_A$, points $P, Q, Q - P$, an integer $m = \sum_{i=0}^{\ell-1} m_i 2^i$.
Assumes: $P, Q - P \notin \{\mathcal{O}, T\}$.
Output: $P + [m]Q$.
$A_{24}^+ \leftarrow (A + 2)/4$
$[X_0 : Z_0]$, $[X_1 : Z_1]$, $[X_2 : Z_2] \leftarrow Q, Q - P, P$
for $i = 0$ to $\ell - 1$ do
    cswap($[X_1 : Z_1]$, $[X_2 : Z_2]$, $m_i$)
    $[X_0 : Z_0]$, $[X_1 : Z_1] \leftarrow$ xDBLADD($[X_0 : Z_0]$, $[X_1 : Z_1]$, $[X_2 : Z_2]$, $A_{24}^+$)
    cswap($[X_1 : Z_1]$, $[X_2 : Z_2]$, $m_i$)
return $[X_2 : Z_2]$
```

The three-point ladder uses a `cswap` routine that performs in constant time a swap of $[X_1 : Z_1]$ with $[X_2 : Z_2]$ if the bit $m_i = 1$. In between swaps, the three-point ladder calls the `xDBLADD` routine, which combines a doubling with a differential addition, the latter being given by the formula

$$[X_{P+Q} : Z_{P+Q}] = \left[4Z_{Q-P}(X_PX_Q - Z_PZ_Q)^2 : 4X_{Q-P}(X_PZ_Q - X_QZ_P)^2\right]. \tag{4}$$

Because each iteration of the loop depends on a secret bit, this is one of the most delicate parts of SIKE, and the target of our first zero-value attack.

2.3 Isogeny evaluation

What time SIKE does not spend performing scalar multiplications, it spends computing and evaluating isogenies. An isogeny is a nonzero group morphism between two elliptic curves. The *degree* of a separable[1] isogeny is the size of its kernel. The task of "computing" an isogeny is customarily decomposed in two steps: During *isogeny computation*, we are given a generator K of a subgroup $\langle K \rangle \subset E_A(\mathbb{F}_{p^2})$, and we want to compute the image curve $E_A/\langle K \rangle$ of the isogeny $\phi_K : E_A \to E_A/\langle K \rangle$ with kernel $\langle K \rangle$, plus possibly some associated data to ϕ_K. Then, during *isogeny evaluation*, we are given E_A, $E_A/\langle K \rangle$, the

[1]The only kind of isogeny relevant in SIKE.

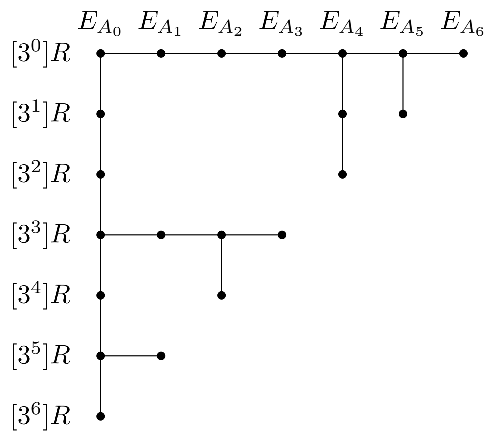


Figure 1: Visualization of a tree traversal game (3^7-isogeny computation).

data associated to ϕ_K, and a point $Q \in E_A$, and we want to compute $\phi_K(Q) \in E_A/\langle K\rangle$. SIKE performs isogeny computations and evaluations for isogenies of degree 2, 3 and 4. Taking degree 3 as example, if we let $K = [X_3 : Z_3]$, SIKE computes the image curve $E_A/\langle K\rangle$ as

$$[A_{24}^- : A_{24}^+] := \big[(3X_3 - Z_3)^3(X_3 + Z_3) \,:\, (3X_3 + Z_3)^3(X_3 - Z_3)\big]; \tag{5}$$

while the map ϕ_K itself is defined as

$$\begin{aligned}\phi_K([X : Z]) \quad = \quad & \big[((X - Z)(X_3 + Z_3) + (X + Z)(X_3 - Z_3))^2 X \\ & : ((X - Z)(X_3 + Z_3) - (X + Z)(X_3 - Z_3))^2 Z\big].\end{aligned} \tag{6}$$

Similar formulas hold for degree 2 and 4 isogenies, see [JAC+20].

After the three-point ladder, the other key step in SIKE is the computation and evaluation of an isogeny of degree 3^{e_3} (or 2^{e_2}). Given a kernel generator K of degree, say, 3^{e_3}, SIKE computes the associated isogeny ϕ_K and, depending on the subroutine, may perform zero or three evaluations of ϕ_K. This is achieved through a combination of 3-isogenies and triplings using a *tree traversal* algorithm guided by what SIKE calls a *strategy*.

A strategy can be modeled as a game which is played on an $n \times n$ rectangular grid, in the area between the axes and above the line $y = x - n$. The goal of the game is to reach the point $(0, n)$ starting from the point $(0, 0)$. To this end, the player can perform two kinds of move: a vertical move downwards (a tripling), or a horizontal move rightwards (a 3-isogeny computation). Moreover, the player can move to any point that was previously visited (every computed point is saved). Downwards moves are unrestricted, but horizontal moves to the right, i.e., from the vertical line $x = i$ to $x = i + 1$, are allowed only after the point $(i, -(n - i))$ was reached (the next curve is computed only after computing the kernel point of the corresponding 3-isogeny). A successfully played game can therefore be represented by a graph of all the visited points, such as Figure 1. Such a graph represents a 3^{e_3}-isogeny computation; the point at $(0, 0)$ represents R. A more formal definition of an isogeny computing strategy can be found in [DFJP14].

2.4 SIKE

SIKE is derived from SIDH, a key exchange scheme, by applying the Hofheinz–Hövelmanns–Kiltz variant of the Fujisaki–Okamoto (FO) transform [HHK17]. SIDH has for public parameters a *starting curve* $E_{A=6}$, a pair of generators P_2, Q_2 of the 2^{e_2}-torsion subgroup of $E_{A=6}$, a pair of generators P_3, Q_3 of the 3^{e_3}-torsion subgroup, and consists of four algorithms: KeyGen_2, KeyGen_3, ShrKey_2 and ShrKey_3. KeyGen_2 generates a key pair $(sk^{(2)}, pk^{(2)})$ by:

1. Picking a random integer $sk^{(2)} \in [0, 2^{e_2} - 1]$.
2. Computing a private kernel generator $R_2 = P_2 + [sk^{(2)}]Q_2$ using the three-point ladder; the way P_2 and Q_2 are set up guarantees that $[2^{e_2-1}]R_2 \neq T$.
3. Computing the private 2^{e_2}-isogeny $\phi_{R_2} : E_{A=6} \to E_{A=6}/\langle R_2 \rangle$ and returning the public key $pk^{(2)} = \big(\phi_{R_2}(P_3), \phi_{R_2}(Q_3), \phi_{R_2}(Q_3 - P_3)\big)$.

KeyGen_3 is identical, except for the facts that the roles of 2 and 3 are swapped, and $sk^{(3)}$ is taken from $[0, 2^{\lfloor \log_2(2^{e_3}) \rfloor} - 1]$.

ShrKey_2 takes as input the public key $pk^{(3)}$, the private key $sk^{(2)}$, and computes a shared key by:

1. Recovering E_A from $pk_{(3)} = (P, Q, Q - P)$.
2. Computing a private kernel generator $S_2 = P + [sk^{(2)}]Q$.
3. Computing the private isogeny $\phi_{S_2} : E_A \to E_A/\langle S_2 \rangle$.
4. Computing the j-invariant of $E_A/\langle S_2 \rangle$ and returning a hash of it.

ShrKey_3 does the same, again swapping the roles of 2 and 3. The properties of isogeny graphs ensure that

$$\mathsf{ShrKey}_2(sk^{(2)}, pk^{(3)}) = \mathsf{ShrKey}_3(sk^{(3)}, pk^{(2)}).$$

SIKE consists of three algorithms: KeyGen, Encaps and Decaps. In *uncompressed* SIKE:

- KeyGen generates a key pair $(sk, pk) \leftarrow \mathsf{KeyGen}_3$.
- Encaps generates an ephemeral key pair $(e, c_0) \leftarrow \mathsf{KeyGen}_2$, derives a shared key $k \leftarrow \mathsf{ShrKey}_2(e, pk)$, and uses it to transport a random message m by sending the ciphertext $ct = (c_0, k \oplus m)$.
- Decaps takes a ciphertext $ct = (c_0, c_1)$, recovers the shared key as $k \leftarrow \mathsf{ShrKey}_3(sk, c_0)$, recovers the message m as $c_1 \oplus k$. The message m is then used both to verify that ct was generated honestly, and to derive a session key.

For efficiency reasons, the roles of 2 and 3 are swapped in *compressed* SIKE.

3 Theoretical analyses

We provide two attacks based on forcing the computing party to evaluate some rational function on an elliptic curve point which has 0 as one of the coordinates, also called *"zero-value point"*. These points are $\mathcal{O} = [1 : 0]$, $T = [0 : 1]$ and the undefined point $[0 : 0]$. The attacking party creates a malicious public key in the sense of KeyGen_2, made of three points $(P, Q, Q - P)$ which are used by the target during Decaps. The first attack is based on forcing the computation of zero-value points during the three-point ladder, and the second attack forces the computation of such points during the isogeny computation. Our final goal is to see where and how such points can occur, and to force a computing party to compute such points based on a secret bit of their private key. The attack is done adaptively in an extend-and-prune manner, that is, we perform the attack in multiple steps; at each step we recover one bit of the private key by assuming that we know the previous parts. We write the secret key as $sk = sk_0 2^0 + sk_1 2^1 + \dots$ and we denote $sk_{<k} = sk_0 2^0 + sk_1 2^1 + \cdots + sk_{k-1} 2^{k-1}$.

3.1 Three-point ladder attack

In this section, we aim to force the apparition of zero-value points $\mathcal{O}$ or T during the computation of the three-point ladder.

A careful analysis of the three-point ladder (as described in Section 2.2) will show that for a triple $(P, Q, Q-P)$ as input, the value of the updated triple after the k^{th} step of the loop is

$$([2^{k+1}]Q, \quad \boldsymbol{-P + [2^{k+1} - sk_{<k}]Q}, \quad P + [sk_{<k}]Q) \quad \text{if } sk_k = 0, \tag{7}$$
$$([2^{k+1}]Q, \quad \boldsymbol{P + [2^{k} + sk_{<k}]Q}, \quad -P + [2^{k} - sk_{<k}]Q) \quad \text{if } sk_k = 1. \tag{8}$$

We show how to create a public key $(Q, P, Q-P)$ such that the second output point (in bold in Equations (7),(8)) is a zero-value point if and only if $sk_k = 0$ (or $sk_k = 1$).

3.1.1 Forcing $\mathcal{O}$

The second output point in Equation (7) is equal to $\mathcal{O}$ if $P = [2^{k+1} - sk_{<k}]Q$. For Equation (8) we must have $P = -[2^k + sk_{<k}]Q$. This simple equation allows us to define the following pair of malicious public key points:

Table 1: Malicious public key points which forces the computation of $\mathcal{O}$

Public key	Q	P	$Q-P$
$pk^{\mathcal{O}}_{k,0}$	Point of order 3^{e_3}	$[2^{k+1} - sk_{<k}]Q$	$-[2^{k+1} - sk_{<k} - 1]Q$
$pk^{\mathcal{O}}_{k,1}$	Point of order 3^{e_3}	$-[2^{k} + sk_{<k}]Q$	$[2^{k} + sk_{<k} + 1]Q$

Note that $P, Q, Q-P \notin \{\mathcal{O}, T\}$, so the public key is well formed. Plugging in $pk^{\mathcal{O}}_{k,0}$ (resp. $pk^{\mathcal{O}}_{k,1}$) in Equation (7) (resp. Equation (8)) shows that on correct guess, the target point becomes $\mathcal{O}$. Furthermore, if the guess is incorrect, then the point is $[2^{k+1} + 2^k]Q$ which has both coordinates non-zero.

3.1.2 Forcing T

The second output point in Equation (7) is equal to T if $P = [2^{k+1} - sk_{<k}]Q + T$. For Equation (8) we must have $P = -[2^k + sk_{<k}]Q - T$. This simple equation allows us to define the following pair of malicious public key points:

Table 2: Malicious public key points which forces the computation of T

Public key	Q	P	$Q-P$
$pk^{T}_{k,0}$	Point of order 3^{e_3}	$[2^{k+1} - sk_{<k}]Q + T$	$-[2^{k+1} - sk_{<k} - 1]Q - T$
$pk^{T}_{k,1}$	Point of order 3^{e_3}	$-[2^{k} + sk_{<k}]Q - T$	$[2^{k} + sk_{<k} + 1]Q + T$

Note that $P, Q, Q-P \notin \{\mathcal{O}, T\}$, so the public key is well formed. Plugging in $pk^{T}_{k,0}$ (resp. $pk^{T}_{k,1}$) in Equation (7) (resp. Equation (8)) shows that on correct guess, the target point becomes T. Furthermore, if the guess is incorrect, then the point is $[2^{k+1} + 2^k]Q$ which has both coordinates non-zero.

3.2 Isogeny computation attack

Our goal is to manipulate the isogeny kernel generating point R of the target party so that it becomes incompatible with isogeny formulas. This leads either to computation of undefined points $[0 : 0]$ (which propagate indefinitely) or to completely (heuristically) random computations. The two cases can be easily distinguished by use of side channels.

We assume that the computing party computes an isogeny of degree 3^{e_3}. The argumentation can be adapted to 2^{e_2} isogenies, or more generally to any set of SIKE parameters.

The isogeny algorithm uses a hard coded strategy, and attempts to compute a 3^{e_3}-isogeny independently of the actual order of the kernel point R. The malicious public key points $(P, Q, Q - P)$, as well as the kernel point R, will be elements of the 2-torsion subgroup $E[2^{e_2}]$ (as opposed to $E[3^{e_3}]$ which is expected by the algorithm). Actually, we will show that there is an exponent $o > 0$ which satisfies so-called *leakage properties*:

L1. If $\operatorname{ord}(R) \big| 2^{o-1}$ then isogenies always lead to undefined points $[0 : 0]$.

L2. If $2^o \big| \operatorname{ord}(R)$ then the isogeny computes random values.

The exponent o depends on the isogeny degree, the tree-traversal strategy, and the order of the target party's point (all being public parameters), and can therefore be precomputed for any set of SIKE parameters.

In this section, we first show how an adversary can control the order of a point in such a way that it depends on the value of a private-key bit. Then we show that there exists an exponent o which satisfies the leakage properties (L1., L2.).

3.2.1 Computing the kernel point

Given any exponent $o > 0$, the goal of the attack is to force the target party to compute a point of order 2^{o-1+sk_k}, sk_k being the value of the private-key bit that we are trying to guess. That is done by creating a public key pk_k^j as shown in Algorithm 2.

Algorithm 2: Malicious public key generation

```
**Input:** Index of bit being guessed $k$, known part of secret key $sk_{<k}$, a public parameter $o$
**Assumes:** $k \leq e_2 - o$
**Output:** Public key $pk_k^j = (P, Q, Q - P)$.
$E \leftarrow$ any supersingular elliptic curve
$P_2, Q_2 \leftarrow$ generators of $E[2^{e_2}]$
Assume $[2^{e_2-1}]Q_2 \neq T$
$S = [2^{e_2-(o-1)}]P_2$
$Q = [2^{e_2-(k+o)}]Q_2$
$P = S - [sk_{<k}]Q$
**return** $pk_k^j = (P, Q, Q - P)$
```

The kernel generating point R obtained from the public key shown in Algorithm 2 satisfies the order constraint as is proved in the following lemma.

Lemma 1. *The kernel generator point $R = P + [sk]Q$ generated from the public key pk_k^j of Algorithm 2 satisfies*

$$\operatorname{ord}(R) = \begin{cases} 2^{o-1} & \text{if } sk_k = 0, \\ 2^o & \text{if } sk_k \neq 0. \end{cases}$$

Proof. Following from Algorithm 2, we have $\operatorname{ord}(S) = 2^{o-1}$ and $\operatorname{ord}(Q) = 2^{k+o}$. Denote with $[sk]Q' = [2^k]Q$, a point of order 2^o. It follows that

$$P + [sk]Q = S + [sk - sk_{<k}]Q = S + [sk_k]Q' + [sk_{k+1}2]Q' + \ldots = S + [sk_k]Q' + Q''$$

where Q'' is a point of order dividing 2^{o-1} and independent from S, by construction. Therefore, if $sk_k = 0$ then $R = S + Q''$ is of order 2^{o-1} because S is of said order, and S and Q'' are independent. On the other side, if $sk_k \neq 0$ then R is of order 2^o because it is the sum of $[sk_k]Q'$ of order 2^o, with $S + Q''$ of order 2^{o-1}. □

3.2.2 Computing the isogeny

In this subsection we will prove the existence of the exponent o which satisfies the leakage properties (L1., L2.).

The 3^{e_3}-isogeny is computed by means of a hard-coded sequence of sub-algorithms which include point tripling, isogeny computation, isogeny evaluation, and saving and loading a point. The order in which these steps are executed are encoded in a strategy as explained in Section 2.3. Because saving or loading a point bears no algebraic structure, we will only analyze the remaining three operations.

The kernel point provided to the isogeny is of incompatible order, which leads to irregular behaviors. During the execution of the 3^{e_3}-isogeny, geometric structure will be lost, and we will essentially work with random points on random elliptic curves. We show when irregular behavior starts, and what types of unexpected behavior can happen.

Point tripling. A point tripling, as can be seen from Equation 3 satisfies the following properties:

- If the curve is degenerate $[A_{24}^+ : A_{24}^-] = [1 : 1]$ and the input point is $\mathcal{O}$ then the output point is undefined $[0 : 0]$.
- $[3]T = T$ even on the degenerate curve.

3-isogeny computation. A 3-isogeny computation, as can be seen from Equation 5 satisfies the following properties:

- If the kernel point is of incorrect order, then the output is an arbitrary elliptic curve which in general is not supersingular and does not share any known geometric relation with the domain curve.
- If the kernel point is $\mathcal{O}$ or T, then the output curve is degenerate $[A_{24}^+ : A_{24}^-] = [1 : 1]$.

3-isogeny evaluation. A 3-isogeny evaluation, as can be seen from Equation 6 satisfies the following properties:

- If the kernel point is of incorrect order, then the image point is arbitrary and does not share any known geometric relation with the preimage point.
- If the input point is equal to the kernel point, the kernel point is $\mathcal{O}$, or the input point is $\mathcal{O}$ then the output is $\mathcal{O}$.
- If the kernel point is $[X : Z]$ and the input is $[Z : X]$, the kernel point is T or the input point is T then the output point is T.
- If both the kernel point and the input point are equal to T then the output is undefined $[0 : 0]$.
- Projectively equivalent points have projectively equivalent images.
- If the image of $[X : Z]$ is $[U : V]$, then the image of $[Z : X]$ is $[V : U]$.

Almost all the analysis of the isogeny computation boils down to analyzing the computations done on the public curve E_A recovered from pk_k^j (i.e., the first branch in the tree traversal). On this curve, the kernel point R is repeatedly tripled, and some intermediate results are saved as $R_i = [3^i]R$ for $i \in I$ where I is a set of indices determined by the strategy. These points are then pushed with the isogeny of kernel $\langle [3^{e_3-1}]R \rangle$. The process is shown in Algorithm 3.

```
Algorithm 3: First vertical branch of the tree-traversal
  Input: Kernel point $R$, starting curve $[A_{24}^{+} : A_{24}^{-}]$, set of indices $I$
  Output: Image curve computed by the first 3-isogeny $[A_{24}^{+}{}' : A_{24}^{-}{}']$,
          Images of points evaluated by the first 3-isogeny $\{\phi_{[3^{e_3-1}]R}([3^i]R)\}_{i\in I}$
  for $i = 0$ to $e_3 - 2$ do
      if $i \in I$ then
          $R_i = R$
      $R = [3]R$
  $[A_{24}^{+}{}' : A_{24}^{-}{}'] =$ curve $E_A/\langle R\rangle$
  for $i \in I$ do
      $R_i = \phi_R(R_i)$
  return $([A_{24}^{+}{}' : A_{24}^{-}{}'], \{R_i\}_{i\in I})$
```

Due to the fact that point R is of incompatible order, the image curve $[A_{24}^{+}{}' : A_{24}^{-}{}']$ is an arbitrary, generally non-supersingular curve. From this point onward, the points and the curves are arbitrary. The only deterministic "structure" that the points can carry is that some of them may have equal projective coordinates. There are three cases to consider.

1. **There is a pair of saved points with equal coordinates.** Assume that R_a and R_b have equal projective coordinates and $a < b \in I$. As the points are equal, their images through consecutive 3-isogenies will stay equal until we compute the isogeny generated by some image of R_b. This isogeny will send R_a to $\mathcal{O}$. The point $\mathcal{O}$ is fixed by point tripling and isogenies. At a certain point an isogeny of kernel $\mathcal{O}$ is computed, whose image curve is the degenerate curve $[1 : 1]$, and all the image points become $\mathcal{O}$. The first next tripling will be a tripling of $\mathcal{O}$ on the degenerate curve whose output is the undefined point $[0 : 0]$. From this point onward all values will be 0, and the final j-invariant will be computed as $0/0$. An example of such computation is given in Appendix A.

2. **There is a pair of saved points with *flipped* coordinates.** Assume $R_a = [x : z]$ and $R_b = [z : x]$ and $a < b \in I$. The property of R_a, R_b having *flipped* coordinates is preserved until the image of R_b is used to compute an isogeny. This isogeny will send the image of R_a to T. The point T is fixed by point tripling and isogenies. At a certain point an isogeny of kernel T is computed, whose image curve is the degenerate curve $[1 : 1]$, and all the image points become T. The first next image of T under the isogeny generated by T is the undefined point $[0 : 0]$. From that point onward all values will be 0, and the final j-invariant will be computed as $0/0$.

3. **There are no points with equal nor *flipped* coordinates.** The points and curves became arbitrary after computing the first 3-isogeny. From this point onward we have different arbitrary values which propagate. The final curve and its j-invariant is random.

Our goal is to force the computing party to fall into case 1 or 2 if $sk_k = 0$ and into case 3 if $sk_k \neq 0$. Note that case 1 is equivalent to the existence of $a < b \in I$ such that $[3^a]R = \pm[3^b]R$, which is similar to case 2 but with $[3^a]R = \pm[3^b]R + T$. In case 3, such $a < b \in I$ simply do not exist. These properties are characterized by the order of point R as shown in the following lemma.

Lemma 2. *For each set of SIKE parameters, there is an integer $o > 0$ such that:*

(A1) $\mathrm{ord}(R) | 2^{o-1}$ *if and only if there exists $a < b \in I$ such that* $[3^a]R = \pm[3^b]R$*,*

(A2) $2^o \big| \operatorname{ord}(R)$ *if and only if there are no such* $a < b \in I$.

Furthermore, if R *is dependent from* T*, then the following is true for the same exponent* o*:*

(B1) $\operatorname{ord}(R) \big| 2^o$ *if and only if there exists* $a < b \in I$ *such that* $[3^a]R = \begin{cases} \pm[3^b]R & \textit{or,} \\ \pm[3^b]R + T, \end{cases}$

(B2) $2^{o+1} \big| \operatorname{ord}(R)$ *if and only if there are no such* $a < b \in I$.

Proof. Let $2^r = \operatorname{ord}(R)$. Then $[3^a]R = \pm[3^b]R$ is equivalent to a modular equation $3^a \mp 3^b \equiv 0 \pmod{2^r}$. Furthermore, an equality modulo 2^r projects to an equality for all divisors of 2^r. Therefore, it is enough to show that there is one exponent r such that $3^a \mp 3^b \not\equiv 0 \pmod{2^r}$ for all $a < b \in I$, since the equation is certainly true for $r = 0$. (A1) and (A2) follow from solving the equation for r by observing the SIKE parameters. We call this exponent o the *break-point exponent.*

If R depends on T, then we must have $[2^{r-1}]R = T$. Therefore the statement $[3^a]R = \pm[3^b]R + T$ is equivalent to $3^a \mp 3^b + 2^{r-1} \equiv 0 \pmod{2^r}$. For each $a, b \in I$ the equations $3^a \mp 3^b \equiv 0 \pmod{2^{r-1}}$ and $3^a \mp 3^b \not\equiv 0 \pmod{2^r}$ are equivalent to $3^a \mp 3^b + 2^{r-1} \equiv 0 \pmod{2^r}$. Together with (A1) and (A2) this proves (B1) and (B2). □

Side-channel attack. Using Lemma 2, two different outcomes of the isogeny computation can be forced depending on the value of a secret bit: the party either computes only zero values from a certain point in the tree traversal and onward, or completely random values. When zero values can be distinguished from random ones with a side channel, such a behavior enables an adaptive bit-by-bit key recovery.

We propose to perform the zero-value distinction on the subroutine responsible of the subfield inversion within the j-invariant computation. This is because the j-invariant computation occurs at one of the last steps of the key exchange, making it a conveniently identifiable target, and because the subfield inversion is usually computed as $a^{-1} = a^{p-2}$; a noticeable sequence of ≥ 200 similar field operations. This scenario is illustrated in Algorithm 4.

Algorithm 4: Attack scenario relating to the isogeny computation

```
Input: Breaking point $o$
Output: The secret key $sk$
for $k = 0$ to $e_2 - o - 1$ do
    Assume we know $sk_{<k} = \sum_{i=0}^{k-1} sk_i 2^i$
    Generate $(P, Q, Q - P)$ with $(k, sk_{<k}, o)$ as in Algorithm 2.
    Make the target party computes $R = P + [sk]Q$.
    if $sk_k = 0$ then
        $0^{p-2}$ is computed.
    else if $sk_k \neq 0$ then
        $random^{p-2}$ is computed.
    Distinguish which one of the two computations above occurred and obtain $sk_k$.
Brute-force the remaining $o$ bits of the secret key.
return $sk$
```

Other parameters. The attack was analyzed in the case of the target computing an isogeny of degree 3^{e_3}. In the general case, the isogeny is of degree ℓ^{e_ℓ} and the cardinality of the curve is of $\ell^{2e_\ell} q^{2e_q}$. The secret key is extracted in base-q digits per turn, $sk = sk_0 q^0 + sk_1 q^1 + \cdots$ and the point R has an order of a power of q. The exponent o can be found with the same procedure as in Lemma 2, where $(3, e_3, 2, e_2)$ is swapped with (ℓ, e_ℓ, q, e_q). In particular, the case with T can never happen unless $q = 2$. We report the values for o for both parties and all parameter sets in Table 3. All attacks are also included on our GitHub page: https://anonymous.4open.science/r/SIKE-SCA-2022-E500.

Table 3: Break-point exponents o for all parameter sizes.

SIKE instances	p434	p503	p610	p751
2^{e_2}-isogeny	3	4	2	5
3^{e_3}-isogeny	9	7	7	8

3.3 Distinguishing zero values

There exist many ways to distinguish zero values in a power or EM trace. For instance, [Gou01, AT03] argue that a zero value can be observed in a single measurement by noticing a significant drop of power consumption or EM radiations. In practice, however, this method requires setting a manual threshold based on observing the measured samples. As a result, to remove the hassle of detecting zero values manually, we propose to make the distinction by comparing a trace (or a collection of traces) to another. Note that such a methodology is aligned with our threat model, as the adversary can query the executions of the targeted cryptographic procedure with inputs that force any kind of computations as many times as required.

t-test. Welch's t-test [SM15] is a statistical hypothesis testing method that examines whether two classes of traces were sampled from indistinguishable populations. In our case, such a test is used to compare a collection of multiple traces of (known to be) nonzero value executions against a collection of multiple traces that may or may not exhibit zero values depending on a secret bit. This is especially useful when the exact instants in time at which the expected zero values occur are unknown; as it is the case with the three-point ladder.

The test proceeds by computing t statistics based on the observed means and variances of two power or EM sample populations. Significant t values imply that the two collections are *not* indistinguishable. When attacking the three-point ladder, some t values are thus expected to be large when a zero is processed in the target trace collection, since zero values are not affected by coordinate randomization. When nonzero values are processed in both classes, all the t values are expected to be small, even when identical inputs were provided to the three-point ladder due to the randomization of coordinates [KAJ17].

Collision power analysis. When the target operation has a known timing and can be forced to process zero and random (i.e., nonzero) values regardless of the value for the private key, a more efficient approach can be employed based on collision power analysis [SWP03, MME10]. In this case, the values processed in a trace are detected by comparing the trace against two baselines (i.e., templates). The two baselines correspond to power traces relating the same execution as the target trace but in which processed values are known to be zero and random. Such a scenario is applicable to the j-invariant computation, as the entire procedure processes zero values when a special input is provided.

Since the baselines are captured from the target device, the overall attack process resembles an online template attack [BCP$^+$14]. The main difference is that an online template attack creates templates as bits are recovered which are then matched against segmented portions of a single target trace, while our attack creates target traces as bits are recovered which are then matched against only two templates (the baselines). As opposed to a regular template attack though, no learning phase (nor programmable device) is ever required, as the template matching is performed by collision power analysis. Also, online template attacks are prevented by coordinate randomization while our attack prevails against this countermeasure.

In practice, a collision power analysis is typically mounted using Pearson's Correlation Coefficient (PCC). This technique correlates a target power trace $\mathrm{Tr} \in \mathbb{R}^m$ with a baseline $\mathrm{B} \in \mathbb{R}^m$ by computing $\rho(\mathrm{Tr}, \mathrm{B}) = \mathrm{Cov}(\mathrm{Tr}, \mathrm{B})/(\sigma_{\mathrm{Tr}}\sigma_{\mathrm{B}})$. The greater the value of the coefficient, the more correlated the trace is to the baseline. As a result, a zero value is detected when the corresponding trace has a greater correlation coefficient with the zero-valued baseline than with the random-valued one.

4 Experimental evaluations

In this section, we verify the correctness of the attacks in practice.

4.1 Setup

4.1.1 Hardware

The experimental evaluations of the attack were performed using two different setups: one for electromagnetic analysis, and the other for power consumption analysis. The equipment, which differs according to the platform, is respectively listed in Table 4.

EM acquisition. The victim board for the electromagnetic side-channel attack is an STMicroelectronics STM32F4DISCOVERY kit with a STM32F407VGT6 microcontroller featuring an ARM Cortex-M4. The low-level hardware library LibOpenCM3[2] was used as in the PQM4: the post-quantum crypto library for the ARM Cortex-M4 [KRSS]. A 168 MHz CPU clock is generated from an 8 MHz quartz oscillator. The host computer communicates with the target via an UART bus. The microcontroller is not decapsulated and the board was not modified. A Langer LF-U 5 passive near-field probe captures the electromagnetic activity of the target during computations. This signal is amplified by a Langer PA303 amplifier and sampled by a Lecroy WaveRunner 640zi oscilloscope. The host computer is able to recover an image of EM emanations from the target by requesting such data to the oscilloscope. The amplifier bandwidth is of 3 GHz but an analog filter at 1 GHz is applied by the oscilloscope before sampling at 2.5 samples per ns.

Table 4: Specific equipment used to measure experimental side-channel leakages.

Description of the electromagnetic platform	
STM32F4DISCOVERY	Victim board, featuring a Cortex-M4 microprocessor.
LF-U 5	Langer near-field probe.
PA303	Langer amplifier of 30 dB gain (up to 3 GHz).
Description of the power consumption platform	
CW308T-STM32F3	Victim board, featuring a Cortex-M4 microprocessor.
CW1173	ChipWhisperer-Lite board, used to exchange with the victim.
CW308 UFO	ChipWhisperer interface board which features a convenient access to the power consumption of the victim.
CW502 LNA	Low-noise amplifier of 20 dB gain (up to 2GHz).

Power acquisition. The experimental evaluation of the power analysis attack was performed using the ChipWhisperer framework [OC14]. The equipment is set up as follows. The CW308T-STM32F3 is plugged onto the CW308 UFO which is itself linked to the CW1173 via a cable of 20 pins. The CW1173 is controlled by the computer which sends

[2]http://libopencm3.org/

commands with arbitrary payloads through a micro-USB cable. The power consumption of the CW308T-STM32F3 is exposed through a shunt resistor on the CW308 UFO board. The voltage drop of the shunt resistor is measured by the WaveRunner 640zi through the CW502 LNA which is connected with optical fibers. In this setup, the WaveRunner 640zi is configured with an analog bandwidth of 200 MHz and a sampling rate of 250 samples per μs. The clock speed of the STM32 is set to 44 MHz.

4.1.2 Software

The attacked software calls the functions of the recommended implementation of SIKEp434 for 32-bit Cortex-M4 microcontrollers with input ciphertexts received from the computer. Such a software enables the acquisition of electromagnetic emanations and power consumption of specific operations during the execution of the SIKE key decapsulation.

The target software codes for electromagnetic and power consumption setups slightly differ. In both codes, the host computer sends a ciphertext to the target, and the target computes the shared secret with the decapsulation procedure using a static private key. For electromagnetic analysis, the implementation is designed from an USART example for STM32F4-DISCOVERY in LibOpenCM3 while, for power analysis, the target runs a custom version of ChipWhisperer's simpleserial library.

Moreover, the scalar multiplication of the library is protected with coordinate randomization. As the original library does not offer such a countermeasure, coordinates are randomized after computing the coefficient of the received curve, and before the Montgomery three-point ladder. A random representation of the points $(Q, P, Q - P)$ is generated from the received affine coordinates (x_Q, x_P, x_{Q-P}). This countermeasure consumes $6 \times \log_2(p)$ random bits to generate three random Z coordinates and requires three $\mathbb{F}_{p^2}$ multiplications.

Finally, the code is further modified to allow a GPIO to trigger the side-channel trace collection of the oscilloscope. When toggled, the GPIO notifies the oscilloscope to start the capture of the electromagnetic activity or power consumption of the STM32. The purpose of such a modification is to make the collection of traces more convenient. Note, however, that the attack is still applicable without a trigger and that the synchronization of traces can be performed using, e.g., cross-correlation techniques [DPN+16].

4.2 Three-point ladder

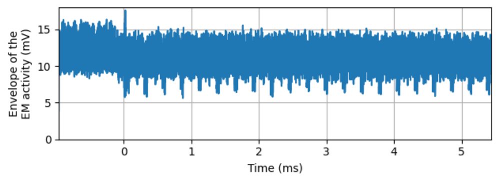
The following sections describe a proof of concept for the attack on the three-point ladder (as described in Section 3.1).

4.2.1 Target operation

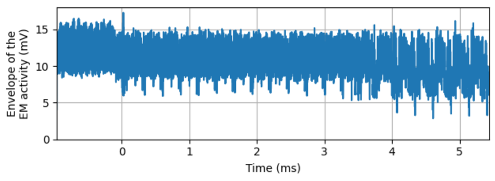
The three-point ladder we plan to attack is located in the decapsulation step, Decaps (see Section 2.4). At the beginning of the decryption, a session key is computed using a static secret key sk and a malicious ciphertext ct_i. More precisely, $P + [sk]Q$ is computed using P, Q and $Q - P$, a malicious triplet $pk^T_{k,0}$ forged as in Section 3.1 such that the zero-value point T appears during the computation when the bit of the secret key we are trying to determine is processed. We want to detect these zero values using the EM emissions of the victim board described in Section 4.1.1.

4.2.2 Experimental procedure

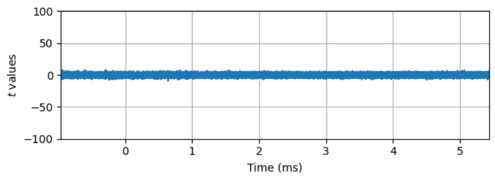
The victim board runs the decapsulation routine, and will thus compute the three-point ladder to get $P + [sk]Q$. The computer sends the triplets with randomized coordinates (see Section 4.1.2) to the board. For the sake of generality, we will present the reasoning



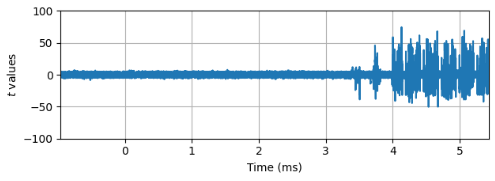
(a) Trace for the wrong hypothesis.



(b) Trace for the correct hypothesis.



(c) t-test for the wrong hypothesis.



(d) t-test for the correct hypothesis.

Figure 2: Experimental results for an attack on one bit.

for a general bit sk_k ($k \geq 0$). To find sk_k, we will compare the electromagnetic emissions of the board performing the ladder computations with three types of input:

- A random, correct triplet of points,
- A malicious triplet $pk^T_{k,0}$ and
- A malicious triplet $pk^T_{k,1}$.

The attacker launches thus campaigns in each of these cases by recording multiple traces of the execution of the three-point ladder on the board. For the sake of diversity, $pk^T_{k,0}$ (respectively $pk^T_{k,1}$) inputs are generated multiple times by choosing different Q points and curves. Then, two t-tests (see Section 3.3) are computed:

T0 between the traces obtained with a random triplet and the ones obtained with $pk^T_{k,0}$,

T1 between the traces obtained with a random triplet and the ones obtained with $pk^T_{k,1}$.

The t-test graph exhibiting peaks corresponds to the correct hypothesis. Comparing t-tests T0 and T1 eliminates the need for a threshold indicating that the values are significant.

Once a bit has been found, the process to get the following bits is similar. Each time, we assume that the previous bits, i.e., the bits at each index 0, 1, …, $k-1$ are known, in order to determine bit sk_k as seen in Section 3.1.

4.2.3 Results

For the experiments, we use the secret $(...10001011110001011011101010011 10)_2$ (in base 2 with the least significant bit on the right). We want to find a general bit, for instance bit 12, knowing the previous bits, $(110101001110)_2$. We will thus record three trace collections in each case as described in Section 4.2.2. Do note that the collection obtained with random inputs can be reused for both T0 and T1, and for all bits.

Figures 2a and 2b show each a trace taken from these collections. The time window is limited to the first twenty loop steps of the three-point ladder. To prevent aliasing issues on these figures, the envelope of the traces is represented instead of the raw trace. The envelope is computed by applying a low-pass filter with a cutoff frequency of 750 kHz on the rectified trace. The trace corresponding to correct random inputs (not shown) is similar to the one obtained for the wrong hypothesis (Figure 2a). The difference between the trace corresponding to the correct bit hypothesis and the wrong one is visible to the naked eye. However, if one wants to automate the attack in the future, it is better to compute a statistical test, as explained in Section 4.2.2. We use 10 raw traces from each collection (so 30 traces per secret bit) to compute T0 and T1 as it is enough. The t values are shown on Figures 2c and 2d. The difference between both t-tests is obvious: we find that 1 is the correct bit hypothesis. The first peak appearing on Figure 2d looks a bit different from the following ones: at the loop step where the bit sk_k is processed and zero coordinates start appearing, they are not necessarily represented by 0, but by $p + \sqrt{-1} \cdot p$, p or $\sqrt{-1} \cdot p$ since $\mathbb{F}_{p^2}$ elements are represented in $[0, 2p-1]^2$ (in the Montgomery domain, see [Gue02]). During the following steps, more and more zeros represented by 0 appear, hence the change in peak appearance.

We thus managed to show that it is possible to recover any bit of the secret key during the three-point ladder computation using appropriate malicious triplets even if projective coordinates are randomized.

4.3 Isogeny computation

The following experiment describes a proof of concept for the attack on the j-invariant computation as described in Section 3.2 using power analysis.

4.3.1 Target operation

In the experiment, only the power consumption of the first field multiplication (i.e., `fpmul_mont` in the source code) from the modular inversion involved in the computation of the j-invariant is measured. As a myriad of zero-valued operations are executed when the j-invariant is zero, it would be superfluous to capture the entire computation of j and compare every operation involved. Still, this specific multiplication was selected because the same function is called a total of 93 times during the modular inversion (with all zero when the j-invariant is zero). Accordingly, in case the leakage of one field multiplication alone is not enough to correctly detect the presence of zero values, a single trace including all the calls to the field multiplication can be segmented into multiple sub-power traces to boost the accuracy of the comparison (even though this practice turned out to be unnecessary in our experiment).

4.3.2 Experiment procedure

The experiment followed the approach with the two baselines as described in Section 3.3. The steps taken by each single experiment in Algorithm 5. Let E be a random curve, P_2, Q_2 generators of $E[2^{e_2}]$ with $[2^{e_2-1}]Q \neq T$, and let n correspond to the number of bits in a private key (i.e., $n = 218$ for SIKEp434) and o to the order corresponding to the breaking point between zero or nonzero j-invariants (i.e., $o = 9$ for SIKEp434).

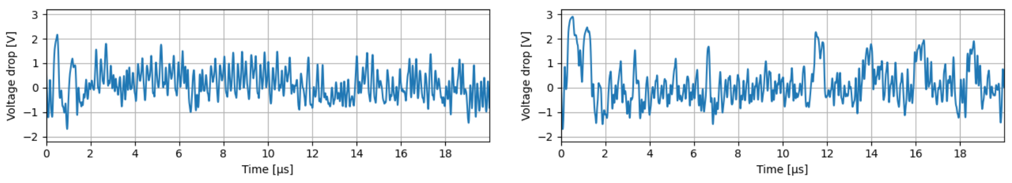
The setup for power consumption acquisition (see Table 4) is used to capture all power traces. An example for the two baselines which both consist of $m = 4,960$ samples is shown in Figure 3.

4.3.3 Results

Across $N = 1,000$ experiments, the first $n - o = 208$ private-key bits were always successfully extracted through collision power analysis with baselines. Table 5 shows the average correlation coefficients when a target trace is compared against the two

Algorithm 5: Experimental attack on isogeny computation (cf. Algorithms 2,4).

```
1 Set up a fixed random private key $sk$ on the STM32.
2 do capture the baselines $\mathrm{B}^{(0)}$, $\mathrm{B}^{(*)}$ for the two categories—zero and random:
3     Send $Q^{(0)} = [2^{e_2-(o-1)}]Q_2$, $P^{(0)} = [2^{e_2-(o-1)}]P_2$, $P^{(0)} - Q^{(0)}$ to capture $\mathrm{B}^{(0)} \in \mathbb{R}^m$.
4     Send $Q^{(*)} = [2^{e_2-(o+1)}]Q_2$, $P^{(*)} = [2^{e_2-(o+1)}]P_2$, $P^{(*)} - Q^{(*)}$ to capture $\mathrm{B}^{(*)} \in \mathbb{R}^m$.
5 for all recoverable bits $0 \leq i < n - o$ (starting with $sk' = 0$) do
6     Send $Q_i = [2^{e_2-(i+o)}]Q_2$, $P_i = [2^{e_2-(o-1)}]P_((2 - [sk']Q_i$, $P_i - Q_i$, and capture
        $\mathrm{Tr}_i \in \mathbb{R}^m$.
7     if $\rho(\mathrm{Tr}_i, \mathrm{B}^{(*)}) > \rho(\mathrm{Tr}_i, \mathrm{B}^{(0)})$ then $sk' = sk' + 2^i$.
8 return $sk'$
```



(a) Zero-valued baseline $\mathrm{B}^{(0)}$. (b) Random-valued baseline $\mathrm{B}^{(*)}$.

Figure 3: Examples of baseline traces corresponding to a single $\mathbb{F}_{p^2}$ multiplication processing zero values in one case, and random (nonzero) values in the other.

baselines. This outcome shows that the recommended implementation of SIKE for Cortex-M4 is vulnerable to the zero-value attack on the j-invariant computation as described in Section 3.2.

Table 5: Average PCCs between baselines and target traces ($N = 1,000$).

	Target	
Baselines	$j = 0$	$j \neq 0$
$j = 0$	**0.9975**	0.3915
$j \neq 0$	0.3916	**0.9909**

Given the significant correlations for a single field multiplication, the results give strong evidence that zero values can be easily detected by comparing the power consumption of an operation with a baseline.

5 Countermeasures

Our attacks rely on ciphertexts containing maliciously generated point triplets (P, Q, R) which are not the legitimate images $(\phi(P_3), \phi(Q_3), \phi(Q_3 - P_3))$ of the public 3^{e_3}-torsion basis under an isogeny of degree 2^{e_2}. As we already mentioned, validating SIKE ciphertexts is a problem believed to be as hard as breaking SIKE itself, thus we cannot hope to completely rule out side-channel attacks using malicious ciphertexts. Nevertheless our malicious ciphertexts deviate from the legitimate format in a detectable way, letting us design an effective countermeasure.

Indeed, the attack on the isogeny computation uses points of order 2^{e_2} instead of 3^{e_3}, while the attack on the three-point ladder uses some points of order $2 \cdot 3^n$. To counter them it is enough to check that P and Q are both of order 3^{e_3}, and that they generate

$E_A[3^{e_3}]$, i.e. that[3] $[3^{e_3-1}]P \neq [\pm 3^{e_3-1}]Q$. Note that by construction we automatically have that $R = Q - P$. We shall name this the *CLN test*, after the names of its first proponents [CLN16].

The CLN test may not be enough to guarantee that the elliptic curve E_A is supersingular, but this can be ensured at little extra cost. Indeed, having checked that P and Q generate $E_A[3^{e_3}]$, we already know that $\#E_A(\mathbb{F}_{p^2}) = 3^{2e_3}D$ for some integer D, and Hasse's bound proves that

$$(p-1)^2 \leq 3^{2e_3}D \leq (p+1)^2, \qquad \text{implying that} \qquad 2^{2e_2} - \frac{4p}{3^{2e_3}} \leq D \leq 2^{2e_2}.$$

Because $3^{2e_3} \approx p$, only a few choices are possible for D, the largest one corresponding to a supersingular curve. Better yet, if we find some power $2^d | D$ such that $2^d > 4p/3^{2e_3}$, then, dividing all sides by 2^d, we conclude that the curve is supersingular.

For all SIKE proposed parameters, except the NIST IV parameter `SIKEp610`, it turns out that $4p/3^{2e_3} < 2$. But any Montgomery curve has order divisible by 4 (see [CS18]), thus we are done. For `SIKEp610` $2(p-2)/3^{2e_3} < 8$, it is thus enough to check that $A+2$ and $A-2$ are both squares in $\mathbb{F}_{p^2}$, which implies that the four 2-torsion points and the four special 4-torsion points are all in E_A, proving that it is supersingular.

Remark 1. Swapping the roles of 2 and 3, analogous checks would also work for verifying public keys. However, $4p/2^{2e_2}$ tends to be quite large for SIKE instances: as much as ≈ 447.6 for `SIKEp751`. It is thus not realistic to look for simple algebraic conditions that would guarantee the existence of points of small order 3^n. Instead, one may take random points on E_A, and multiply them by a cofactor until a point of sufficiently large order 3^n is found. Alternatively, one may be just content with testing that $2^{2e_2} | \#E_A(\mathbb{F}_{p^2})$: although this does not guarantee that the curve is supersingular, it is believed to be computationally hard to find ordinary curves with such a large fixed factor.

This also applies to compressed SIKE, where the roles of the 2^{e_2}- and 3^{e_3}-torsion are swapped. The *entangled basis generation* procedure of [ZSP+18] guarantees that $2^{2e_2} | \#E_A(\mathbb{F}_{p^2})$,[4] and ciphertext decompression ensures that (P, Q) generates $E_A[2^{e_2}]$. Implementations may then choose between relying on a computational assumption ensuring that E_A is supersingular, or doing a little extra work to find a point of appropriate order 3^n on E_A. At any rate, our attacks do not apply to compressed SIKE because of this.

Remark 2. Following [GPST16], one could check the condition $e_{3^{e_3}}(P, Q) = e_{3^{e_3}}(P_3, Q_3)^{2^{e_2}}$, which is more stringent than the one enforced by the CLN test. However, this check is still not enough to verify SIDH keys (precisely the point made in [GPST16]), it is expensive, and its effectiveness against side-channel attacks is dubious. We thus advise against it.

We added the countermeasure to Microsoft's PQCrypto-SIDH library[5] and tested it on a laptop equipped with an Intel Kaby Lake i7-8650U CPU clocked at 1.90GHz with Turbo Boost switched off. We benchmarked both the generic C version and the version with x64 assembly optimizations. The results are reported in Table 6 and show a performance hit of around 10%.

5.1 Bypassing the countermeasure

In a work predating the first round of NIST's competition, Koziel, Azarderaksh, and Jao explored zero-value attacks on SIDH/SIKE protected with coordinate randomization and

[3]Equivalent conditions are that R is also of order 3^{e_3}, or that the Weil pairing $e_{3^{e_3}}(P, Q)$ has multiplicative order 3^{e_3}.

[4]The proof therein requires that $A^2 - 4$ is a square in $\mathbb{F}_{p^2}$, thus an implementation of compressed SIKE expecting malicious ciphertexts should check this condition before generating the basis.

[5]https://github.com/microsoft/PQCrypto-SIDH

Table 6: Performance in cycles of plain SIKE decapsulation vs SIKE decapsulation with CLN countermeasure.

	SIKEp434	SIKEp503	SIKEp610	SIKEp751
Optimized x64	11,533,938	15,860,919	30,854,816	49,661,481
Optimized x64 w/ CLN	12,991,717	17,801,396	34,414,507	54,291,130
ratio	1.1264	1.1223	1.1154	1.0932
Optimized generic	105,686,298	161,183,747	324,716,783	543,154,342
Optimized generic w/ CLN	117,307,201	180,086,020	356,924,412	607,925,540
ratio	1.1100	1.1173	1.0992	1.1193

the CLN test [KAJ17]. They presented four attacks and claimed that these manage to bypass the countermeasure. We shall now explain why none of them applies to SIKE.

The first attack ([KAJ17, §5]) targets the three-point ladder. It is based on the fact that, in the main loop of the ladder, one of the inputs to the differential addition is either P or $Q - P$, according to a secret bit. First, we note that the attack assumes the coordinates of P and $Q - P$ are not randomized before entering the ladder, but in this case a simpler DPA would work as well. Second, since the introduction of a more efficient ladder in [FHLOJRH18], all implementations of SIDH, including SIKE, have moved away from using a fixed pair of points in the loop, as can be seen in Algorithm 1.

The second attack ([KAJ17, §6]) also targets the three-point ladder, and asks to find "[Montgomery] curves with points $P_0 = (\pm 1, y)$ with a large order". However, as explained in Section 2, such points always have order 4, the attack can thus not be mounted against any version of SIDH implemented using Montgomery curves, such as SIKE.

The third attack ([KAJ17, §7.2]) is the most enticing. The idea is to make a guess on the secret, and then generate a ciphertext such that, if the guess is correct, the private isogeny computed during decapsulation will pass through the (supersingular) curve with $A = 0$. This is clearly feasible, and although the ciphertext generated would not pass the Fujisaki–Okamoto test, such an attack is hard to detect at an early phase. The only obstacle, already realized by [KAJ17], is that in all efficient implementations of SIDH/SIKE the curve E_A is internally represented as $(A_{24}^{+} : C_{24})$ or as $(A_{24}^{+} : A_{24}^{-})$. Neither of these encodings can produce a zero if it represents an elliptic curve, thus it seems that this idea cannot be used in a realistic scenario.

The last attack ([KAJ17, §7.3]) tries to force zeros in points pushed through the isogeny computation and, in a sense, foreshadowed our attack presented in Section 3.2. The description in [KAJ17] is however incomplete and, by the authors' own admission, does not apply to Montgomery curves.

It appears thus that, to the present date, there is no known zero-value attack that would bypass the CLN countermeasure, although the third attack in [KAJ17] is a close call. Whether the CLN test can block all types of zero-value attacks is something that could be investigated using, e.g., the techniques of [SCDJB21]. We leave this question for future work.

6 Conclusion

We described two zero-value attacks against SIKE: one on the three-point ladder, the other on the isogeny computation. Both attacks are based on special-point inputs that enable an adaptive bit-by-bit key recovery. We analyzed them in theory, but also verified them experimentally on the recommended SIKE implementation for Cortex-M4 with both EM and power analysis using different techniques. At last, we argued that the Costello–Longa–Naehrig test which verifies the order of the points is sufficient to stop the attacks.

Countless similar attacks are likely to exist, however finding a zero-value or other side-channel attack capable of bypassing both coordinate randomization and the CLN countermeasure remains an open question. Relatedly, future work could also determine whether zero-value points are possible on the compressed variant of SIKE. Finally, novel formulas for both the three-point ladder and the isogeny computation designed to resist zero-value attacks are another interesting direction to explore.

References

[Apo20] Daniel Apon. Passing the final checkpoint! NIST PQC 3rd round begins, 2020. https://meetings.ams.org/math/fall2020se/meetingapp.cgi/Paper/1656,https://www.scribd.com/document/474476570/PQC-Overview-Aug-2020-NIST.

[AT03] Toru Akishita and Tsuyoshi Takagi. Zero-value point attacks on elliptic curve cryptosystem. In Colin Boyd and Wenbo Mao, editors, *ISC 2003*, volume 2851 of *LNCS*, pages 218–233. Springer, Heidelberg, October 2003.

[BCP+14] Lejla Batina, Lukasz Chmielewski, Louiza Papachristodoulou, Peter Schwabe, and Michael Tunstall. Online template attacks. In Willi Meier and Debdeep Mukhopadhyay, editors, *INDOCRYPT 2014*, volume 8885 of *LNCS*, pages 21–36. Springer, Heidelberg, December 2014.

[BDE+18] Jonathan Bootle, Claire Delaplace, Thomas Espitau, Pierre-Alain Fouque, and Mehdi Tibouchi. LWE without modular reduction and improved side-channel attacks against BLISS. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part I*, volume 11272 of *LNCS*, pages 494–524. Springer, Heidelberg, December 2018.

[BDK+21] Michiel Van Beirendonck, Jan-Pieter D'anvers, Angshuman Karmakar, Josep Balasch, and Ingrid Verbauwhede. A side-channel-resistant implementation of SABER. *Journal on Emerging Technologies in Computing Systems*, 17(2), apr 2021.

[BP18] Leon Groot Bruinderink and Peter Pessl. Differential fault attacks on deterministic lattice signatures. *IACR TCHES*, 2018(3):21–43, 2018. https://tches.iacr.org/index.php/TCHES/article/view/7267.

[CCD+21] Pierre-Louis Cayrel, Brice Colombier, Vlad-Florin Dragoi, Alexandre Menu, and Lilian Bossuet. Message-recovery laser fault injection attack on the classic McEliece cryptosystem. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part II*, volume 12697 of *LNCS*, pages 438–467. Springer, Heidelberg, October 2021.

[CLN16] Craig Costello, Patrick Longa, and Michael Naehrig. Efficient algorithms for supersingular isogeny Diffie-Hellman. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 572–601. Springer, Heidelberg, August 2016.

[CMP18] Laurent Castelnovi, Ange Martinelli, and Thomas Prest. Grafting trees: A fault attack against the SPHINCS framework. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018*, pages 165–184. Springer, Heidelberg, 2018.

[Cor99] Jean-Sébastien Coron. Resistance against differential power analysis for elliptic curve cryptosystems. In Çetin K. Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems*, pages 292–302, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

[Cos19] Craig Costello. Supersingular isogeny key exchange for beginners. In Kenneth G. Paterson and Douglas Stebila, editors, *SAC 2019*, volume 11959 of *LNCS*, pages 21–50. Springer, Heidelberg, August 2019.

[Cos21] Craig Costello. The case for SIKE: a decade of the supersingular isogeny problem. Cryptology ePrint Archive, Report 2021/543, 2021.

[CS18] Craig Costello and Benjamin Smith. Montgomery curves and their arithmetic - the case of large characteristic fields. *Journal of Cryptographic Engineering*, 8(3):227–240, September 2018.

[DF17] Luca De Feo. Mathematics of isogeny based cryptography, 2017.

[DFJP14] Luca De Feo, David Jao, and Jérôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *Journal of Mathematical Cryptology*, 8(3):209–247, 2014.

[DPN$^+$16] Margaux Dugardin, Louiza Papachristodoulou, Zakaria Najm, Lejla Batina, Jean-Luc Danger, and Sylvain Guilley. Dismantling real-world ECC with horizontal and vertical template attacks. In François-Xavier Standaert and Elisabeth Oswald, editors, *COSADE 2016*, volume 9689 of *LNCS*, pages 88–108. Springer, Heidelberg, April 2016.

[EFGT17] Thomas Espitau, Pierre-Alain Fouque, Benoît Gérard, and Mehdi Tibouchi. Side-channel attacks on BLISS lattice-based signatures: Exploiting branch tracing against strongSwan and electromagnetic emanations in microcontrollers. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 1857–1874. ACM Press, October / November 2017.

[FHLOJRH18] Armando Faz-Hernández, Julio López, Eduardo Ochoa-Jiménez, and Francisco Rodríguez-Henríquez. A faster software implementation of the supersingular isogeny Diffie-Hellman key exchange protocol. *IEEE Transactions on Computers*, 67(11):1622–1636, 2018.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 537–554. Springer, Heidelberg, August 1999.

[FO13] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. *Journal of Cryptology*, 26(1):80–101, January 2013.

[GJN20] Qian Guo, Thomas Johansson, and Alexander Nilsson. A key-recovery timing attack on post-quantum primitives using the Fujisaki-Okamoto transformation and its application on FrodoKEM. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 359–386. Springer, Heidelberg, August 2020.

[GLK21] Aymeric Genêt, Natacha Linard de Guertechin, and Novak Kaluđerović. Full key recovery side-channel attack against ephemeral SIKE on the Cortex-M4. In Fabrizio Bhasin, Shivam and De Santis, editor, *Constructive Side-Channel Analysis and Secure Design*, pages 228–254, Cham, 2021. Springer International Publishing.

[Gou01] Louis Goubin. A sound method for switching between Boolean and arithmetic masking. In Çetin Kaya Koç, David Naccache, and Christof Paar, editors, *CHES 2001*, volume 2162 of *LNCS*, pages 3–15. Springer, Heidelberg, May 2001.

[Gou03] Louis Goubin. A refined power-analysis attack on elliptic curve cryptosystems. In Yvo Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 199–210. Springer, Heidelberg, January 2003.

[GPST16] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 63–91. Springer, Heidelberg, December 2016.

[Gue02] Shay Gueron. Enhanced montgomery multiplication. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 46–56. Springer, 2002.

[GV18] Steven D. Galbraith and Frederik Vercauteren. Computational problems in supersingular elliptic curve isogenies. *Quantum Information Processing*, 17(10):265, 2018.

[GW17] Alexandre Gélin and Benjamin Wesolowski. Loop-abort faults on supersingular isogeny cryptosystems. In Tanja Lange and Tsuyoshi Takagi, editors, *Post-Quantum Cryptography - 8th International Workshop, PQCrypto 2017*, pages 93–106. Springer, Heidelberg, 2017.

[HHK17] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the Fujisaki-Okamoto transformation. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 341–371. Springer, Heidelberg, November 2017.

[IT03] Tetsuya Izu and Tsuyoshi Takagi. Exceptional procedure attack on elliptic curve cryptosystems. In Yvo Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 224–239. Springer, Heidelberg, January 2003.

[JAC$^+$20] David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, Vladimir Soukharev, David Urbanik, Geovandro Pereira, Koray Karabina, and Aaron Hutchinson. SIKE. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[JD11] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011*, pages 19–34. Springer, Heidelberg, November / December 2011.

[KAJ17] Brian Koziel, Reza Azarderakhsh, and David Jao. Side-channel attacks on quantum-resistant supersingular isogeny Diffie-Hellman. In Carlisle Adams and Jan Camenisch, editors, *SAC 2017*, volume 10719 of *LNCS*, pages 64–81. Springer, Heidelberg, August 2017.

[KL19] Juliane Krämer and Mirjam Loiero. Fault attacks on UOV and Rainbow. In Ilia Polian and Marc Stöttinger, editors, *COSADE 2019*, volume 11421 of *LNCS*, pages 193–214. Springer, Heidelberg, April 2019.

[KPHS18] Philipp Koppermann, Eduard Pop, Johann Heyszl, and Georg Sigl. 18 seconds to key exchange: Limitations of supersingular isogeny Diffie-Hellman on embedded devices. Cryptology ePrint Archive, Report 2018/932, 2018. https://eprint.iacr.org/2018/932.

[KRSS] Matthias J. Kannwischer, Joost Rijneveld, Peter Schwabe, and Ko Stoffelen. PQM4: Post-quantum crypto library for the ARM Cortex-M4. https://github.com/mupq/pqm4.

[LNPS20] Norman Lahr, Ruben Niederhagen, Richard Petri, and Simona Samardjiska. Side channel information set decoding using iterative chunking - plaintext recovery from the “classic McEliece” hardware reference implementation. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part I*, volume 12491 of *LNCS*, pages 881–910. Springer, Heidelberg, December 2020.

[MGTF19] Vincent Migliore, Benoît Gérard, Mehdi Tibouchi, and Pierre-Alain Fouque. Masking Dilithium - efficient implementation and side-channel evaluation. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *ACNS 19*, volume 11464 of *LNCS*, pages 344–362. Springer, Heidelberg, June 2019.

[MME10] Amir Moradi, Oliver Mischke, and Thomas Eisenbarth. Correlation-enhanced power analysis collision attack. In Stefan Mangard and François-Xavier Standaert, editors, *CHES 2010*, volume 6225 of *LNCS*, pages 125–139. Springer, Heidelberg, August 2010.

[Moo18] Dustin Moody. Let’s get ready to rumble - The NIST PQC "competition", 2018. https://csrc.nist.gov/presentations/2018/let-s-get-ready-to-rumble-the-nist-pqc-competiti.

[OC14] Colin O’Flynn and Zhizhang (David) Chen. ChipWhisperer: An open-source platform for hardware embedded security research. In Emmanuel Prouff, editor, *COSADE 2014*, volume 8622 of *LNCS*, pages 243–260. Springer, Heidelberg, April 2014.

[PP21] Peter Pessl and Lukas Prokop. Fault attacks on CCA-secure lattice KEMs. *IACR TCHES*, 2021(2):37–60, 2021. https://tches.iacr.org/index.php/TCHES/article/view/8787.

[PSKH18] Aesun Park, Kyung-Ah Shim, Namhun Koo, and Dong-Guk Han. Side-channel attacks on post-quantum signature schemes based on multivariate quadratic equations. *IACR TCHES*, 2018(3):500–523, 2018. https://tches.iacr.org/index.php/TCHES/article/view/7284.

[RHHM17] Melissa Rossi, Mike Hamburg, Michael Hutter, and Mark E. Marson. A side-channel assisted cryptanalytic attack against QcBits. In Wieland Fischer and Naofumi Homma, editors, *CHES 2017*, volume 10529 of *LNCS*, pages 3–23. Springer, Heidelberg, September 2017.

[SBWE20] Okan Seker, Sebastian Berndt, Luca Wilke, and Thomas Eisenbarth. SNI-in-the-head: Protecting MPC-in-the-head protocols against side-channel analysis. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020*, pages 1033–1049. ACM Press, November 2020.

[SCDJB21] Vladimir Sedlacek, Jesús-Javier Chi-Domínguez, Jan Jancar, and Billy Bob Brumley. A formula for disaster: A unified approach to elliptic curve special-point-based attacks. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021*, pages 130–159, Cham, 2021. Springer International Publishing.

[SM15] Tobias Schneider and Amir Moradi. Leakage assessment methodology - A clear roadmap for side-channel evaluations. In Tim Güneysu and Helena Handschuh, editors, *CHES 2015*, volume 9293 of *LNCS*, pages 495–513. Springer, Heidelberg, September 2015.

[SRSWZ21] Thomas Schamberger, Julian Renner, Georg Sigl, and Antonia Wachter-Zeh. A power side-channel attack on the CCA2-Secure HQC KEM. In Pierre-Yvan Liardet and Nele Mentens, editors, *Smart Card Research and Advanced Applications*, pages 119–134, Cham, 2021. Springer International Publishing.

[SWP03] Kai Schramm, Thomas J. Wollinger, and Christof Paar. A new class of collision attacks and its application to DES. In Thomas Johansson, editor, *FSE 2003*, volume 2887 of *LNCS*, pages 206–222. Springer, Heidelberg, February 2003.

[TDFEMP21] Élise Tasso, Luca De Feo, Nadia El Mrabet, and Simon Pontié. Resistance of isogeny-based cryptographic implementations to a fault attack. In Shivam Bhasin and Fabrizio De Santis, editors, *Constructive Side-Channel Analysis and Secure Design*, pages 255–276, Cham, 2021. Springer International Publishing.

[Tho17] Erik Thormarker. *Post-Quantum Cryptography: Supersingular Isogeny Diffie-Hellman Key Exchange.* Thesis, Stockholm University, 2017.

[Ti17] Yan Bo Ti. Fault attack on supersingular isogeny cryptosystems. In Tanja Lange and Tsuyoshi Takagi, editors, *Post-Quantum Cryptography - 8th International Workshop, PQCrypto 2017*, pages 107–122. Springer, Heidelberg, 2017.

[UJ20] David Urbanik and David Jao. New techniques for SIDH-based NIKE. *Journal of Mathematical Cryptology*, 14(1):120–128, 2020.

[UXT$^+$22] Rei Ueno, Keita Xagawa, Yutaro Tanaka, Akira Ito, Junko Takahashi, and Naofumi Homma. Curse of re-encryption: A generic power/EM analysis on post-quantum KEMs. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 296–322, 2022.

[XIU$^+$21] Keita Xagawa, Akira Ito, Rei Ueno, Junko Takahashi, and Naofumi Homma. Fault-injection attacks against NIST's post-quantum cryptography round 3 KEM candidates. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 33–61. Springer, 2021.

[XPSR$^+$21] Zhuang Xu, Owen Michael Pemberton, Sujoy Sinha Roy, David Oswald, Wang Yao, and Zhiming Zheng. Magnifying side-channel leakage of lattice-based cryptosystems with chosen ciphertexts: The case study of Kyber. *IEEE Transactions on Computers*, 2021.

[ZSP$^+$18] Gustavo Zanon, Marcos A. Simplício Jr., Geovandro C. C. F. Pereira, Javad Doliskani, and Paulo S. L. M. Barreto. Faster isogeny-based compressed key agreement. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018*, pages 248–268. Springer, Heidelberg, 2018.

[ZYD$^+$20] Fan Zhang, Bolin Yang, Xiaofei Dong, Sylvain Guilley, Zhe Liu, Wei He, Fangguo Zhang, and Kui Ren. Side-channel analysis and countermeasure design on arm-based quantum-resistant SIKE. *IEEE Transactions on Computers*, 69(11):1681–1693, 2020.

A A visual explanation of the isogeny attack

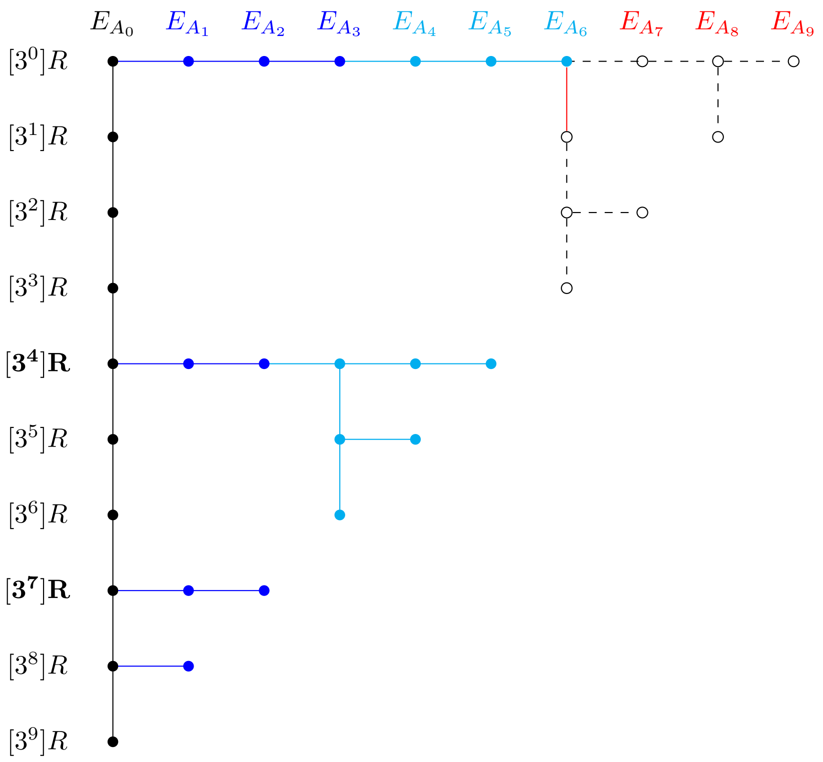


Figure 4: An example of an isogeny computation with a kernel of wrong order

In Figure 4, we can see a 3^{10}-isogeny computation with a kernel of incompatible order.

- With black we denote regular points and supersingular elliptic curves.
- With blue we denote arbitrary points, isogenies with bad kernel, and arbitrary (non-supersingular) elliptic curves.
- With cyan we denote the point $\mathcal{O}$, isogenies with kernel $\langle\mathcal{O}\rangle$, triplings of $\mathcal{O}$ and degenerate elliptic curves.
- With red we denote the isogeny (tripling) which first creates the undefined point $[0 : 0]$
- With circles we denote undefined points $[0 : 0]$.
- With dashed lines we denote isogenies which send points to the undefined point and undefined elliptic curves.

Assume that the points $[3^4]R$ and $[3^7]R$ are equal. On the first curve E_{A_0} the point R is tripled 9 times, and the points $[3^0]R, [3^4]R, [3^7]R$ and $[3^8]R$ are saved. A 3-isogeny is computed from $[3^9]R$, and the saved points are pushed. The images of $[3^4]R$ and $[3^7]R$ are still equal. Another 3-isogeny is computed from the image of $[3^8]$ and the saved points are pushed. The images of $[3^4]R$ and $[3^7]R$ are still equal. The third 3-isogeny sends the image of $[3^4]R$ to $\mathcal{O}$. The next isogeny, generated by $\mathcal{O}$ sends all saved points to $\mathcal{O}$. The same is true of the next two isogenies. The first next tripling is the tripling of $\mathcal{O}$ on the degenerate curve $[1 : 1]$ which outputs the undefined point $[0 : 0]$. From this point onward all the outputs are $[0 : 0]$.