# Fault attack on SQIsign

Jeonghwan Lee$^{1[0009-0004-0957-5051]}$, Donghoe Heo$^{1[0000-0001-9300-2985]}$, Hyeonhak Kim$^{1[0000-0003-2782-6999]}$, Gyusang Kim$^{1[0000-0002-9937-674X]}$, Suhri Kim$^{3[0000-0003-2665-7142]}$, Heeseok Kim$^{2[0000-0001-8137-4810]}$, and Seokhie Hong$^{\star 1[0000-0001-7506-4023]}$

1 School of Cybersecurity, Korea University, Seoul 02841, South Korea
2 Department of AI Cyber Security, College of Science and Technology, Korea University, Sejong 30019, South Korea
3 School of Mathematics, Statistics and Data Science, Sungshin Women's University, Seoul 02844, South Korea

**Abstract.** In this paper, we introduce the first fault attack on SQIsign. By injecting a fault into the ideal generator during the commitment phase, we demonstrate a meaningful probability of inducing the generation of order $\mathcal{O}_0$. The probability is bounded by one parameter, the degree of commitment isogeny. We also show that the probability can be reasonably estimated by assuming uniform randomness of a random variable, and provide empirical evidence supporting the validity of this approximation. In addition, we identify a loop-abort vulnerability due to the iterative structure of the isogeny operation. Exploiting these vulnerabilities, we present key recovery fault attack scenarios for two versions of SQIsign—one deterministic and the other randomized. We then analyze the time complexity and the number of queries required for each attack. Finally, we discuss straightforward countermeasures that can be implemented against the attack.

**Keywords:** Isogeny · Quaternion Algebra · Post-Quantum Cryptography · Fault Attack.

## 1 Introduction

National Institute of Standards and Technology (NIST) initiated the Post Quantum Cryptography (PQC) standardization process in 2016, responding to the significant threat to widely used public-key encryption systems due to quantum computers. After several rounds of evaluation, CRYSTALS-Kyber in Key Encapsulation Mechanism (KEM) category, and CRYSTALS-Dilithium, FALCON, and SPHINCS+ in digital signature category were selected as schemes for standardization on July 5th, 2022. Subsequently, in August 2023, NIST announced the first drafts for CRYSTALS-Kyber, CRYSTALS-Dilithium, and SPHINCS+ as FIPS 203, 204, and 205, respectively (standardization for Falcon is still in progress). However, because three of the four standardization schemes

$^\star$ Corresponding author.

were lattice-based, NIST began an additional standardization process for digital signatures based on various mathematical problems. As of June 2023, 40 additional submissions have been published, and security analyses are ongoing for each algorithm.

SQIsign[1] is the only isogeny-based digital signature among the candidates in the additional round 1. In particular, it is currently recognized as the most compact post-quantum signature scheme with efficient verification. SQIsign effectively exploits the Deuring correspondence, which establishes a mathematical equivalence between the domain of supersingular elliptic curves and that of maximal orders in a quaternion algebra. Unlike many isogeny-based schemes similar to SIDH that are vulnerable to Castryck-Decru-Maino-Martindale-Robert attacks [7,17,18], SQIsign remains secure, as its public information does not disclose torsion point information.

In addition to ensuring the mathematical security of cryptographic algorithms, it is necessary to consider aspects such as memory usage, key size, ciphertext or signature size, and optimized implementation to meet the needs of various cryptographic infrastructures. Equally important is the need to address the resilience of these schemes against physical attacks, namely, side-channel and fault attacks. Substantial research has been done on physical attacks on isogeny-based cryptography, especially fault attacks. Notable examples include the introduction of loop-abort faults during isogeny computation in the SIDH cryptosystem [15], resulting in the exposure of secret keys. A similar vulnerability for CSIDH is presented in a related paper [5]. Safe-error key recovery attacks on both SIKE and CSIDH have been demonstrated by [6], which perturbs memory locations on the computation of isogeny. Furthermore, the fault attack introduced by [2] manipulates the computation related to the *orientation* of a point in CSIDH, causing the secret-dependent faulty curve from which the attack can recover the secret key. Vulnerabilities have also been discovered outside of isogeny computation. The authors of [20] and [19] have shown that they can inject faults into torsion points in SIDH and exchange them for vulnerable torsion points, enabling an attacker to retrieve a secret key. Injecting faults into intermediate curves in SIDH [1] have been introduced, which can leak information about certain aspects of secret isogeny over the $\mathbb{F}_p$ curve.

However, to the best of our knowledge, there have been no reported fault attacks on SQIsign, especially in the context of quaternion algebra operations. In this paper, we present the first fault attack on SQIsign. First, we prove that by injecting a fault into the ideal generator, we can, with a significant probability, generate the order $\mathcal{O}_0$ instead of the expected commitment ideal during the commitment phase. The probability is bounded by a single factor: the degree of a commitment isogeny. We show that the probability can be roughly estimated by assuming the uniform randomness of a random variable. The validity of this approximation is further supported by empirical estimation. We also reveal the existence of a loop-abort vulnerability caused by the iterative structure of the

[1] Throughout the paper, the term "SQIsign" and its implementation refer to version 1.0 released as a NIST Round 1 addition signature candidate on June 1, 2023.

isogeny operation, as similarly described in [15]. Using the identified vulnerabilities, we present key recovery attack scenarios for two versions of SQIsign[2], one deterministic, and the other randomized. Then, we analyze the time complexity and the number of queries required for each attack. Finally, we discuss simple countermeasures that can be taken against the attack.

The rest sections of the paper are structured as follows: Section 2 provides the mathematical background necessary for understanding the SQIsign signing process, along with an introduction to fault injection. In Section 3, the paper analyzes two vulnerabilities induced by fault injection in the commitment phase. Section 4 presents key recovery attack scenarios for deterministic and randomized SQIsign based on the earlier vulnerability analyses. Finally, Section 5 discusses countermeasures and concludes the paper.

## 2 Preliminaries

In this paper, we consider a prime number $p$ that satisfies $p \equiv 3 \pmod 4$, and $\mathbb{F}_{p^2}$ denotes a finite field of size $p^2$. The mathematical aspects of SQIsign are outlined in Section 2.1. The signing process of SQIsign, which is the focus of the attack, is explained in Section 2.2. And Section 2.3 introduces two types of fault vulnerabilities in SQIsign. More details can be found in [8,10,11].

### 2.1 Mathematical background

**Elliptic curves and isogenies.** SQIsign makes use of supersingular elliptic curves over $\mathbb{F}_{p^2}$ with an endomorphism ring defined over the same field. It uses a *separable* isogeny, completely described by its kernel. Using formulas such as Vélu[21] or $\sqrt{\text{elu}}$[4], the separable isogeny $\phi$ can be computed from its kernel $G$, denoted as $\phi : E \to E/G$. The degree of the composite isogeny $\phi \circ \psi$ is equal to $deg(\phi) \cdot deg(\psi)$. It can be factored into the composition of $e_i$ isogenies of degree $p_i$ for $i = 1$ to $n$ for any *separable* isogeny $\phi$ of degree $d = \prod_{i=1}^{n} p_i^{e_i}$, satisfying $\phi \circ \hat{\phi} = [deg(\phi)]$, the multiplication-by-$[deg(\phi)]$ map on $E_1$, vice versa.

All operations in SQIsign are conducted in Montgomery form using only the x-coordinate for efficiency, and higher extension fields than $\mathbb{F}_{p^2}$ are not used. However, computing isogenies with large degrees using $(p^2 - 1)$-torsion points is infeasible over $\mathbb{F}_{p^2}$. To address this, SQIsign utilizes two curves: an elliptic curve $E$ with $(p+1)$-torsion and its twist $E^t$ with $(p-1)$-torsion. Isogenies are computed by forming two kernels from these curves. Despite the use of two kernels, the computations in $E$ and $E^t$ are the same in Montgomery form. Therefore, in this paper, we express them under a single elliptic curve $E$. For instance, when

[2] While SQIsign [8] does not explicitly mention deterministic and randomized versions, it is noted that Fiat-Shamir heuristic-based digital signatures such as CRYSTALS-Dilithium [12] and HAETAE [9] (one of the candidates on NIST round 1 additional signatures) included or now include deterministic algorithms. Thus, it is reasonable to consider the attack on the two versions of SQIsign.

representing the set of torsion points, the set of $(p+1)$-torsion points on $E$ is denoted as $E[p+1]$, and the set of $(p-1)$-torsion set on $E^t$ is also denoted as $E[p-1]$.

**Quaternion algebras.** SQIsign uses the quaternion algebra over $\mathbb{Q}$, denoted as $B_{p,\infty}$. This algebra is based on $1, i, j, k$, where $i^2 = -1$, $j^2 = -p$, and $k = ij = -ji$. A canonical involution in $B_{p,\infty}$ maps an element $\alpha = a_1 + a_2 i + a_3 j + a_4 k$ to its conjugate $\overline{\alpha} = a_1 - a_2 i - a_3 j - a_4 k$. The *reduced trace* and *reduced norm* are defined as $tr(\alpha) = \alpha + \overline{\alpha}$ and $nrd(\alpha) = \alpha\overline{\alpha}$, respectively. An element $\alpha$ in $B_{p,\infty}$ is called integral if both its *reduced trace* and *reduced norm* belong to $\mathbb{Z}$.

A full-rank $\mathbb{Z}$-lattice in $B_{p,\infty}$ is termed a fractional ideal $I$, and can be expressed as $I = \alpha_1\mathbb{Z} + \alpha_2\mathbb{Z} + \alpha_3\mathbb{Z} + \alpha_4\mathbb{Z}$. An order $\mathcal{O}$ is a subring of $B_{p,\infty}$ that also qualifies as a fractional ideal. Elements of order $\mathcal{O}$ are integral as they have *reduced trace* and *reduced norm* in $\mathbb{Z}$. An element $\alpha \in \mathcal{O}$ can be denoted as $[\alpha]_B$, which is a 4-dimensional coordinate of $g$ with respect to the basis $B$ of $\mathcal{O}$. A maximal order is not encompassed by any larger order. The left order of a fractional ideal $I$ is denoted as $\mathcal{O}_L(I) = \{\alpha \in B_{p,\infty} \mid \alpha I \subset I\}$, and the right order is represented as $\mathcal{O}_R(I)$. An order $\mathcal{O}$ can define a left fractional $\mathcal{O}$-ideal if the set $I$ is closed under left-multiplication by elements of $\mathcal{O}$, which implies $\mathcal{O} \subseteq \mathcal{O}_L(I)$. A right fractional $\mathcal{O}'$-ideal is formed when $I$ is closed under right-multiplication by elements of $\mathcal{O}'$. An $I$ that qualifies as both a left fractional $\mathcal{O}$-ideal and a right fractional $\mathcal{O}'$-ideal is called a connecting $(\mathcal{O}, \mathcal{O}')$-ideal.

A fractional ideal that is contained in both its left and right orders is called an integral ideal, or simply an ideal. The norm of an ideal $I$, represented as $nrd(I)$, is defined as $\gcd(\{nrd(\alpha) \mid \alpha \in I\})$. A left $\mathcal{O}$-ideal $I$ can be generated as $I = \mathcal{O} \cdot \alpha + \mathcal{O} \cdot nrd(I)$, provided $\alpha \in \mathcal{O}$ and $\gcd(nrd(\alpha), nrd(I)^2) = nrd(I)$, and similarly for a right $\mathcal{O}'$-ideal. In short, this notation can be written as $\mathcal{O} \cdot \alpha + \mathcal{O} \cdot N = \mathcal{O}\langle \alpha, N \rangle$, applicable to any order $\mathcal{O}$ and integer $N$, simplifying the expression. The product $IJ$ of ideals, where $\mathcal{O}_R(I) = \mathcal{O}_L(J)$, is the ideal generated by the products of pairs in $I \times J$. This product is also an integral ideal, and its left order is $\mathcal{O}_L(IJ) = \mathcal{O}_L(I)$, while its right order is $\mathcal{O}_R(IJ) = \mathcal{O}_R(J)$. The conjugate of an ideal $\overline{I}$ is the set of conjugates of elements of $I$, which is an ideal satisfying $I\overline{I} = nrd(I)\mathcal{O}_L(I)$ and $\overline{I}I = nrd(I)\mathcal{O}_R(I)$ when $I$ is invertible. SQIsign uses only *locally principal* ideals, which are always invertible.

**The Deuring correspondence.** In the domain of quaternion algebra and elliptic curves, a bijective correspondence is established. An isogeny, denoted as $\phi : E \to E^{'}$, is linked with an ideal $I_\phi$. This ideal has a left order $\mathcal{O}$, which is isomorphic to the endomorphism ring of $E$, denoted as $End(E)$, and a right order $\mathcal{O}'$, which is isomorphic to the endomorphism ring of $E'$, denoted as $End(E')$. Moreover, the degree of the isogeny $\phi$ is the same as the norm of the ideal $I_\phi$. The dual isogeny, denoted as $\widehat{\phi}$, corresponds to the conjugate of the ideal $I_\phi$, denoted as $\overline{I_\phi}$, which is equal to the ideal of the dual isogeny, denoted as $I_{\widehat{\phi}}$. The composition of two isogenies, denoted as $\phi \circ \psi$, corresponds to the multiplication of two ideals $I_\psi \cdot I_\phi$. Lastly, when two ideals, $I$ and $J$, are equivalent, their right orders are isomorphic. Therefore, the isogenies associated with these ideals, $\phi_I$ and $\phi_J$, have isomorphic codomains.

**The KLPT algorithm.** The KLPT algorithm [16] is a method that can be used to find an equivalent ideal $J$ with a prescribed norm, given an initial ideal $I$. This algorithm can be used to efficiently determine the isogeny between elliptic curves with known endomorphism rings. However, the original formulation of the KLPT algorithm is only applicable to $\mathcal{O}_0$-ideals, where $\mathcal{O}_0$ represents a conditioned maximal order. The authors of [10] later generalized the algorithm to accommodate arbitrary order $\mathcal{O}$. This generalization expands the applicability of the KLPT algorithm, but requires a connecting $(\mathcal{O}_0, \mathcal{O})$-ideal and a 1.5 times larger norm bound for the output than the previous one.

**Computational probelms.** The foundation of SQIsign is based on two computational problems: the *isogeny path problem* and the *endomorphism ring problem*. The *isogeny path problem* is about finding an isogeny $E_1 \to E_2$ given two elliptic curves $E_1$ and $E_2$. The *endomorphism ring problem* is about computing $End(E)$ given an elliptic curve $E$. For supersingular case, these problems are equal under polynomial time reductions [13,22].

**Table 1.** SQIsign parameters

| **NIST-I** |
|---|
| $p =$ 0x34e29e286b95d98c33a6a86587407437252c9e49355147ffffffffffffffffff |
| $f =$ 75 |
| $T = 3^{36} \cdot 7^4 \cdot 11 \cdot 13 \cdot 23^2 \cdot 37 \cdot 59^2 \cdot 89 \cdot 97 \cdot 101^2 \cdot 107 \cdot 109^2 \cdot 131 \cdot 137 \cdot 197^2 \cdot 223 \cdot 239$ $\cdot 383 \cdot 389 \cdot 491^2 \cdot 499 \cdot 607 \cdot 743^2 \cdot 1033 \cdot 1049 \cdot 1193 \cdot 1913^2 \cdot 1973$ |
| **NIST-III** |
| $p =$ 0x3df6eeeab0871a2c6ae604a45d10ad665bc2e0a90aeb751c722f669356ea4684c6174 c1ffffffffffffffffffffffff |
| $f =$ 97 |
| $T = 3^{68} \cdot 5 \cdot 7^{12} \cdot 11^4 \cdot 13 \cdot 47^4 \cdot 89 \cdot 113 \cdot 157^4 \cdot 173 \cdot 233 \cdot 239 \cdot 241 \cdot 443 \cdot 509^4 \cdot 569$ $\cdot 761^4 \cdot 1229 \cdot 2393 \cdot 3371 \cdot 4517 \cdot 5147 \cdot 5693 \cdot 5813 \cdot 9397 \cdot 26777 \cdot 39679 \cdot 47441$ |
| **NIST-V** |
| $p =$ 0x255946a8869bc68c15b0036936e79202bdbe6326507d01fe3ac5904a0dea65faf0a29 a781974ce994c68ada6e1ffffffffffffffffffffffffffffffffffff |
| $f =$ 145 |
| $T = 3^{72} \cdot 5 \cdot 7 \cdot 13^6 \cdot 17 \cdot 37 \cdot 41^6 \cdot 53 \cdot 67^6 \cdot 73 \cdot 103^6 \cdot 127 \cdot 151 \cdot 461^6 \cdot 643 \cdot 733 \cdot 739$ $\cdot 827^6 \cdot 1009 \cdot 2539 \cdot 4153 \cdot 5059 \cdot 7127 \cdot 10597 \cdot 13591 \cdot 14923 \cdot 15541 \cdot 15991 \cdot 18583$ $\cdot 23227 \cdot 48187 \cdot 63247 \cdot 65521 \cdot 318233$ |

## 2.2 SQIsign

We now describe the parameters of SQIsign, and then explain SQIsign signing process, the targeted algorithm in the paper.

**Parameters of SQIsign.** In SQIsign, a value denoted as $T$ is carefully chosen to balance security and computational efficiency. This value determines the available torsion and subsequently influences the degree of the isogenies used in the scheme. Specifically, $T$ is an odd, smooth integer larger than $p^{5/4}$ and satisfies $T|(p^2-1)$. Let $g$ be the largest integer that is a power of 3 and a divisor of $T$. The degree of the commitment isogeny, denoted as $D_{com}$, is then given by $T/3^g$, and the degree of the challenge isogeny, denoted as $D_{chall}$, is given by $2^f 3^g$. The parameter values corresponding to NIST levels are outlined in Table 1.

**Algorithm 1** SQISign signing protocol(simplified)

```
Input: A Message $M$
Input: A secret key $sk$
Output: A signature $\sigma$
    Commitment process
 1: Sample $a_0, b_0$ randomly from interval $[1, D_{com}]$ until $gcd(a_0, b_0, D_{com}) = 1$.
 2: $I_{com} := \mathcal{O}_0\langle\gamma(a_0 + b_0\bar{\theta}), D_{com}\rangle$
 3: $K_{com} := a_0P_0 + b_0\theta(P_0)$, where $P_0 \in B_{0,D_{com}}$
 4: Compute an isogeny $\phi_{com} : E_0 \to E_1$ with the kernel $\langle K_{com}\rangle$.
 5: $P_{1,chall} := \phi_{com}(P_{0,chall}), Q_{1,chall} := \phi_{com}(P_{0,chall})$
    Challenge process
 6: $a_1, b_1 := H(M, E_1)$
 7: Find a $D_{chall}$-torsion basis $(P_{E_1}, Q_{E_1})$ on $E_1$ deterministically.
 8: $K_{chall} := a_1P_{E_1} + b_1Q_{E_1}$
 9: Compute an isogeny $\phi_{chall} : E_1 \to E_2$ with the kernel $\langle K_{chall}\rangle$.
10: Find integers $\alpha, \beta$ such that $[\alpha]P_{1,chall} + [\beta]Q_{1,chall} = K_{chall}$.
11: Compute a left $\mathcal{O}_0$-ideal $I'_{chall}$ corresponding to the isogeny with the kernel
    $\langle[\alpha]P_{0,chall} + [\beta]Q_{0,chall}\rangle$.
12: $I_{chall} := [I_{com}]_* I'_{chall}$
13: Given $K_{chall}$, deterministically find another $D_{chall}$-torsion point $L_{chall}$ such that
    a pair $(K_{chall}, L_{chall})$ is the basis of $E_1[D_{chall}]$.
14: $L'_{chall} := \phi_{chall}(L_{chall})$
15: Find a $D_{chall}$-torsion basis $(P_{E_2}, Q_{E_2})$ on $E_2$ deterministically.
16: Find integers $s_1, s_2$ such that $[s_1]P_{E_2} + [s_2]Q_{E_2} = L'_{chall}$.
17: Given $(L'_{chall}, s_1, s_2)$, deterministically find another $D_{chall}$-torsion point $M'_{chall}$
    such that a pair $(L'_{chall}, M'_{chall})$ is the basis of $E_2[D_{chall}]$.
18: Compute an isogeny $\widehat{\phi_{chall}} : E_2 \to E_1$ with the kernel $\langle L'_{chall}\rangle$.
19: $M_{chall} := \widehat{\phi_{chall}}(M'_{chall})$
20: Find $r$ such that $K_{chall} = [r]M_{chall}$.
    Response process
21: $J_{res} := \overline{I_{secret}} \cdot I_{com} \cdot I_{chall}$
22: Find an ideal $J'_{res}$ of norm $2^e$ which is equivalent to $J_{res}$ such that $J\overline{I_{chall}}$ is cyclic
    using $SigningKLPT$.
23: Convert the ideal $J'_{res}$ into the isogeny $\sigma$ of degree $2^e$ using $sk$.
24: return $\sigma, r, s_1, s_2$
```

**The SQIsign signing process.** The SQIsign signing process consists of three stages: the commitment phase, the challenge phase, and the response phase. Details that are not crucial to the overall understanding of the attack, such as the Montgomery normalized process, are omitted. For a more in-depth algorithmic explanation, refer to [8].

In the commitment phase of SQIsign, the signing oracle initiates by choosing $a_0$ and $b_0$ at random such that $\gcd(a_0, b_0, D_{com}) = 1$. This selection is used to form a commitment ideal $I_{com}$ with a norm of $D_{com}$. Subsequently, the isogeny $\phi_{com} : E_0 \to E_1$, which corresponds to $I_{com}$, is computed. The phase concludes with the evaluation of $P_{0,chall}$ and $Q_{0,chall}$, the basis of $E_0[D_{chall}]$, through $\phi_{com}$ to yield $P_{1,chall}$ and $Q_{1,chall}$. Moving to the challenge phase, the oracle computes the challenge isogeny $\phi_{chall} : E_1 \to E_2$, with a kernel $\langle K_{chall} \rangle$ that is derived by hashing the message $M$ and the elliptic curve $E_1$. The $K_{chall}$ is then decomposed along the basis $P_1$ and $Q_1$, which was obtained during the commitment phase. SQIsign then computes $(s_1, s_2)$, compression of the dual of the challenge isogeny, by computing the kernel generator $L'_{chall}$ of $\widehat{\phi_{chall}}$ and decomposing it along a deterministically computed $D_{chall}$-torsion basis on the curve $E_2$. At the end of the challenge phase, computations are carried out to determine a generator $M_{chall}$ of $\phi_{chall}$, which is obtained by evaluating $M'_{chall}$ through the dual of $\phi_{chall}$. Subsequently, a scalar $r$ is computed, which fulfills the condition $K_{chall} = [r]Q$. In the response phase, the oracle computes $\overline{I_{secret}} \cdot I_{com} \cdot I_{chall}$ and identifies an equivalent ideal $J$ with a norm of $2^e$ using $SigningKLPT$. This process of finding $J$ is repeated until $J\overline{I_{secret}}$ becomes cyclic. SQIsign finally returns $(\sigma, r, s_1, s_2)$ as the completed signature.

Throughout the paper, SQIsign is referred to as deterministic when the initial step of selecting $a_0, b_0$ is performed deterministically. As a result, the protocol will always yield the same output for the same message and key. Conversely, when we mention randomized SQIsign, it implies that these values are determined based on a random number generator, leading to different outputs for each query, even when the same message and key are used.

## 2.3 Fault attacks

Fault attacks are a category of side-channel attacks in which an attacker gains access to a cryptographic device and injects faults, such as power, electromagnetic waves, or laser-induced faults, at specific points. The attacker then analyzes the output to extract sensitive information. Several types of vulnerabilities can be induced by fault attacks, including variable randomization and loop-abort, which we will use in this paper. Variable randomization is random changes induced by faults in the values of variables used in cryptographic computations, resulting in faulty output values. The attacker can extract confidential information by analyzing the values themselves or by comparing them to normal output values [20,2,19]. When it comes to loop abort, it can be done in one of two ways. The first is through variable randomization, as explained earlier. In this approach, the attacker uses fault injection to randomize the loop index, causing the loop to abort at the desired point. The second is instruction skipping. With this, the attacker injects a fault during the execution of a jump instruction that causes the loop to abort. In the field of isogeny-based cryptography, the occurrence of loops as a result of isogeny operations has led to studies of loop-abort fault attacks [15,3]. A similar vulnerability can also be found in SQIsign. We will discuss it in Section 3.

Consider a scenario where the attacker can introduce $n$ number of faults into a cryptographic system while it is in operation. This scenario is known as an $n$-th order fault attack. In Section 4, we will examine a first-order fault attack on the deterministic SQIsign algorithm and a second-order fault attack on the randomized SQIsign algorithm.

## 3 Fault vulnerabilities in SQIsign

In this section, we examine the implementation of SQIsign and analyze its vulnerabilities. First, we look at the vulnerabilities that arise during an ideal generation and the associated operations. After that, we will discuss the characteristics of the isogeny operations and the vulnerabilities that arise from them.

### 3.1 Randomization while generating commitment ideal

**Algorithm 2** Making primitive then generating an ideal $I_{com}$

```
Input: A special $p$-extremal maximal order $\mathcal{O}_0$
Input: A generator coordinates $[g]_B$ w.r.t. the standard basis
Input: The norm $D_{com}$
Output: A commitment ideal $I_{com}$
1: Let $B := \{1, i, j, ij\}$ be a set of the standard basis in quaternion algebra.
2: Let $B_{\mathcal{O}_0} := \{\beta_1, \beta_2, \beta_3, \beta_4\}$ be a set of the basis of order $\mathcal{O}_0$
3: Convert $[g]_B$ into $[g]_{B_{\mathcal{O}_0}}$ using gaussian ellimination.
4: $content := gcd([g]_{B_{\mathcal{O}_0}})$
5: $[g']_{B_{\mathcal{O}_0}} := [g]_{B_{\mathcal{O}_0}} / content$
6: $[g']_B := [g']_{B_{\mathcal{O}_0}} * B_{\mathcal{O}_0}$
7: $D'_{com} := D_{com} / gcd(content, D_{com})$
8: $I_{com} := \mathcal{O}_0 \langle g', D'_{com} \rangle$
9: return $I_{com}$
```

In the SQIsign protocol, the generation of the commitment ideal involves first sampling $a_0$ and $b_0$ in a way that satisfies $\gcd(a_0, b_0, D_{com}) = 1$, as shown in line 1 of Algorithm 1. These values are then used to generate the ideal $I_{com}$. During this process, the generator $\gamma$ of an ideal of norm $D_{com}$ is multiplied by $a_0 + b_0 \cdot \overline{\theta}$ to produce the kernel generator $g = \gamma(a_0 + b_0 \cdot \overline{\theta})$. However, since $g$ may not be primitive, it is essential to convert the input generator to a primitive element before generating the ideal. This conversion is described in Algorithm 3, lines 1 to

```c
int quat_lattice_contains_without_alg(quat_alg_coord_t *coord,
    const quat_lattice_t *lat, const quat_alg_elem_t *x){
    // Test if rank 4 lattice under HNF ...
    // Convert the basis by using gaussian elimination
    // Final test
    // Copy result
    if(res && (coord != NULL)){
        for(int i = 0; i < 4;i++){
            ibz_copy(&((*coord)[i]),&(work_coord[i]));
        }
    }
    // Finalize
}
```

**Fig. 1.** Reference C source code for line 3 in Algorithm 2

8. First, the input generator, represented on the standard basis $B = \{1, i, j, k\}$, is transformed by Gaussian elimination into the basis $B_{\mathcal{O}_0} = \{1, i, \frac{i+j}{2}, \frac{1+k}{2}\}$ of $\mathcal{O}_0$, resulting in $[g]_{B_{\mathcal{O}_0}}$. Then, the gcd of each coordinate of $[g]_{B_{\mathcal{O}_0}}$, called *content*, is computed. This *content* is then used to compute the primitive generator $g'$ by dividing $[g]_{B_{\mathcal{O}_0}}$. $D'_{com}$ is computed as $D_{com}$ divided by $\gcd(content, D_{com})$. Finally, $I_{com}$ is generated by the generator $g'$ and $D'_{com}$.

It is important to note that in the C implementation corresponding to line 3 of Algorithm 2, a copy function is used during the return of the transformed coordinates $[g]_{B_{\mathcal{O}_0}}$. Injecting a fault into the copy process could cause one of the four returned coordinates to be randomized. Assume that a fault is injected during the first iteration of the for loop, and let us denote the faulty copy as $g^f$. Since the fault only causes the coordinates to be randomized with respect to the basis of order $\mathcal{O}_0$, $g^f$ is still an element of $\mathcal{O}_0$. According to Algorithm 2, $g^f$ is then transformed into a primitive element, which leads to the generation of the fault-induced ideal $I^f_{com}$. This process can lead to a vulnerability that can be exploited in a key-recovery-attack scenario in Section 4.

To elaborate further, as described in Section 2, for a $\mathcal{O}_0$-ideal, the generator $\alpha \in \mathcal{O}_0$ must meet the condition $gcd(nrd(\alpha), nrd(I)^2) = nrd(I)$. However, by Theorem 1, the result of the ideal generation process will be $\mathcal{O}_0$ if $\alpha$ is subjected to fault injection to satisfy a certain condition.

**Theorem 1.** *Let $\mathcal{O}_0$ be a maximal order given the basis $(1, i, \frac{i+j}{2}, \frac{1+k}{2})$. For $N \in \mathbb{Z}^+$, $\alpha \in \mathcal{O}_0$, if $\gcd(nrd(\alpha), N) = 1$, then $\mathcal{O}_0 \cdot \alpha + \mathcal{O}_0 \cdot N = \mathcal{O}_0$*

*Proof.* The element $\mathcal{O}_0$ represents an order, and therefore, a ring. Additionally, both $\mathcal{O}_0 \cdot \alpha$ and $\mathcal{O}_0.N$ are principal ideals on the ring $\mathcal{O}_0$. For any arbitrary $\alpha \in \mathcal{O}_0$ expressed as $\alpha = \alpha_1 + \alpha_2 \cdot i + \alpha_3 \cdot \frac{i+j}{2} + \alpha_4 \cdot \frac{1+k}{2}$, the conjugate is given by $\overline{\alpha} = \alpha_1 - \alpha_2 \cdot i - \alpha_3 \cdot \frac{i+j}{2} + \alpha_4 \cdot \frac{1-k}{2}$.

Now, consider $\alpha^{'} = \alpha_1 - \alpha_2 \cdot i - \alpha_3 \cdot \frac{i+j}{2} - \alpha_4 \cdot \frac{1+k}{2} \in \mathcal{O}_0$. Since $\alpha_4 \in \mathbb{Z}$, it follows that $\overline{\alpha} = \alpha^{'} + \alpha_4 \in \mathcal{O}_0$. Consequently, the conjugate $\overline{\alpha}$ of any element $\alpha$ in $\mathcal{O}_0$ is also an element of $\mathcal{O}_0$.

This implies that $nrd(\alpha) = \overline{\alpha} \cdot \alpha \in (\mathcal{O}_0 \cdot \alpha + \mathcal{O}_0 \cdot N)$. Meanwhile, as $\gcd(nrd(\alpha), N) = 1$, there exist integers $a$ and $b$ such that $a \cdot nrd(\alpha) + b \cdot N = 1$. Hence, $\mathcal{O}_0 \cdot \alpha$ and $\mathcal{O}_0 \cdot N$ are relatively prime ideals, and their sum is the entire ring $\mathcal{O}_0$, that is, $\mathcal{O}_0 \cdot \alpha + \mathcal{O}_0 \cdot N = \mathcal{O}_0$.

□

In addition, using Lemma 1, Lemma 2, and Theorem 2, we can compute upper and lower bounds, as well as an approximation, for the probability that the result of the ideal generation process becomes $\mathcal{O}_0$ due to fault injection, based solely on the parameter $D_{com}$. In the following Lemmas and Theorem, $\varphi$ denotes Euler's totient function.

**Lemma 1 (Upper Bound).** *Consider an input $g$ for Algorithm 2, and let $g^f$ be the version of $g$ after fault injection. Suppose that the fault is injected during the first iteration of for-loop, the parameter $D_{com}$ is not divided by 2, and the amount of change $\Delta$ due to fault injection is uniformly random in $\mathbb{Z}$. Then, the probability that $nrd(g^f)$ and $D_{com}$ are coprime has an upper bound:*

$$P(\gcd((nrd(g^f), D_{com})) = 1) < \frac{\varphi(D_{com})}{D_{com}}$$

*Proof.* According to line 2 of Algorithm 1, $g$ is computed as $\gamma \cdot (a_0 + b_0\overline{\theta})$ where $a_0$ and $b_0$ are sampled integers, and $\gamma$ and $\overline{\theta}$ are fixed parameters. Here, when expressing the parameters $\gamma$ and $\overline{\theta}$ in terms of the standard basis of quaternion algebra, each coordinate is represented by an integer value. To be more specific, $\gamma = 1 + \gamma_2 \cdot i + \gamma_3 \cdot j + \gamma_4 \cdot k$, where $\gamma_2, \gamma_3, \gamma_4 \in \mathbb{Z}$, and $\overline{\theta} = \overline{\theta}_1 + \overline{\theta}_2 \cdot i + \overline{\theta}_3 \cdot j + \overline{\theta}_4 \cdot k$, where $\overline{\theta}_1, \overline{\theta}_2, \overline{\theta}_3, \overline{\theta}_4 \in \mathbb{Z}$. Hence, $g$ is represented as $g = g_1 + g_2 \cdot i + g_3 \cdot j + g_4 \cdot k$ where $g_1, g_2, g_3, g_4 \in \mathbb{Z}$. Then, the fault-injected version $g^f$ can be represented as $(g_1 + \Delta) + g_2 \cdot i + g_3 \cdot j + g_4 \cdot k$. According to the properties of the greatest common divisor, we can express $\gcd(nrd(g^f), D_{com})$ as:

$$\gcd(nrd(g^f) - D_{com} \cdot h, D_{com}) = \gcd(nrd(g^f) - nrd(g), D_{com})$$

where $nrd(g) = D_{com} \cdot h$ for some integer $h$. By calculating this in terms of $\Delta$, we get:

$$\begin{aligned}
&\gcd(nrd(g^f) - nrd(g), D_{com}) \\
&= \gcd\left(((g_1 + \Delta)^2 + g_2^2 + g_3^2 \cdot p + g_4^2 \cdot p) - (g_1^2 + g_2^2 + g_3^2 \cdot p + g_4^2 \cdot p), D_{com}\right) \\
&= \gcd\left(\Delta \cdot (\Delta + 2 \cdot g_1), D_{com}\right)
\end{aligned}$$

If $\gcd\left(\Delta \cdot (\Delta + 2 \cdot g_1), D_{com}\right) = 1$, then it implies $\gcd(\Delta, D_{com}) = 1$. Therefore, we can conclude that

$$P(\gcd\left(\Delta \cdot (\Delta + 2 \cdot g_1), D_{com}\right) = 1) < P(\gcd(\Delta, D_{com}) = 1) = \frac{\varphi(D_{com})}{D_{com}}$$

□

**Lemma 2 (Lower Bound).** *Under the same conditions as in Lemma 1, the probability that $nrd(g^f)$ and $D_{com}$ are coprime has a lower bound:*

$$\left(\frac{\varphi(D_{com})}{D_{com}}\right)^3 < P(\gcd((nrd(g^f), D_{com})) = 1)$$

*Proof.* Let us maintain the notations as defined in Lemma 1. To prove the lower bound, we first decompose $P(\gcd(\Delta \cdot (\Delta + 2 \cdot g_1), D_{com}) = 1)$ as follows:

$$P(\gcd(\Delta \cdot (\Delta + 2 \cdot g_1), D_{com}) = 1)$$

$$= P(\gcd(\Delta, D_{com}) = 1 \wedge \gcd(\Delta + 2 \cdot g_1, D_{com}) = 1)$$

$$= \sum_{\substack{\Delta \equiv k \pmod{D_{com}} \\ k \in \mathbb{Z}^*_{D_{com}}}} P(\Delta \equiv k \pmod{D_{com}}) \cdot P(gcd(k + 2 \cdot g_1, D_{com}) = 1)$$

$$= \frac{1}{D_{com}} \cdot \sum_{\substack{\Delta \equiv k \pmod{D_{com}} \\ k \in \mathbb{Z}^*_{D_{com}}}} P(gcd(k + 2 \cdot g_1, D_{com}) = 1)$$

In this context, the probability $P(\gcd(k + 2 \cdot g_1, D_{com}) = 1)$ is equivalent to the probability that $k + 2 \cdot g_1 \pmod{D_{com}} \in \mathbb{Z}^*_{Dcom}$ for the random variables $g_1$. Note that since $g$ is determined by the product of $\gamma$ and $a_0 + b_0\overline{\theta}$, $g_1$ is not uniformly random. Consequently, the distribution of $k + 2 \cdot g_1$ on $\mathbb{Z}_{Dcom}$ is not uniformly random.

In order to examine this distribution, we express $g_1$ in terms of $a_0$ and $b_0$ as $a_0 + m \cdot b_0$ for some $m \in \mathbb{Z}$. By Algorithm 1, $a_0$ and $b_0$ are randomly selected with uniform probability within the specified range: $1 \leq a_0 \leq D_{com}$ and $1 \leq b_0 \leq D_{com}$. Subsequently, they are verified if the selected $a_0$ and $b_0$ satisfy the condition $\gcd(a_0, b_0, D_{com}) = 1$. Only pairs that meet this condition are retained and considered sample values.

Let $S$ denote the set of valid pairs $(a_0, b_0)$ defined as follows.

$$S = \{(a_0, b_0) | 1 \leq a_0 \leq D_{com}, 1 \leq b_0 \leq D_{com}, \gcd(a_0, b_0, D_{com}) = 1\}$$

Then, the probability $P(\gcd(k + 2 \cdot g_1, D_{com}) = 1)$ can be expressed as:

$$P(\gcd(k + 2 \cdot g_1, D_{com}) = 1) = \frac{|\{(a_0, b_0) | (k + 2(a_0 + mb_0)) \in \mathbb{Z}^*_{D_{com}}\}|}{|S|}$$

The set $\{(a_0, b_0) | (k + 2(a_0 + mb_0)) \in \mathbb{Z}^*_{D_{com}}\}$ can be partitioned into subsets $\{(a_0, b_0) | (k+2(a_0+mb_0)) = t\}$ for $t \in \mathbb{Z}^*_{D_{com}}$. Moreover, each subset $\{(a_0, b_0) | (k+2(a_0 + mb_0)) = t\}$ can be further divided into two partitions:

$$\begin{aligned} S_{k,t,1} = \{(a_0, b_0) | 1 \leq a_0, b_0 \leq D_{com}, gcd(b_0, D_{com}) = 1, \\ (k + 2(a_0 + mb_0)) = t\}, \end{aligned}$$

$$S_{k,t,2} = \{(a_0, b_0) | 1 \leq a_0, b_0 \leq D_{com}, gcd(b_0, D_{com}) \neq 1, \\ \gcd(a_0, b_0, D_{com}) = 1, (k + 2(a_0 + mb_0)) = t\}$$

For every $b_0$ satisfies $1 \leq b_0 \leq D_{com}$ and $\gcd(b_0, D_{com}) = 1$, there is a unique value for $a_0$ since $1 \leq a_0 \leq D_{com}$ and $D_{com}$ is coprime to 2. We can deduce that $S_{k,t,1} = \varphi(D_{com})$. Thus,

$$\begin{aligned} &|\{(a_0, b_0) | (k + 2(a_0 + mb_0)) \in \mathbb{Z}^*_{D_{com}}\}| \\ &= \sum_{t \in \mathbb{Z}^*_{D_{com}}} |\{(a_0, b_0) | (k + 2(a_0 + mb_0)) = t\}| \\ &= \sum_{t \in \mathbb{Z}^*_{D_{com}}} (S_{k,t,1} + S_{k,t,2}) > \sum_{t \in \mathbb{Z}^*_{D_{com}}} S_{k,t,1} \\ &= \sum_{t \in \mathbb{Z}^*_{D_{com}}} \varphi(D_{com}) = (\varphi(D_{com}))^2 \end{aligned}$$

Since $D^2_{com} > |S|$,

$$\begin{aligned} &P(\gcd(\Delta \cdot (\Delta + 2 \cdot g_1), D_{com}) = 1) \\ &= \frac{1}{D_{com}} \cdot \sum_{\substack{\Delta \equiv k \pmod{D_{com}} \\ k \in \mathbb{Z}^*_{D_{com}}}} P(gcd(k + 2 \cdot g_1, D_{com}) = 1) \\ &= \frac{1}{D_{com}} \cdot \sum_{\substack{\Delta \equiv k \pmod{D_{com}} \\ k \in \mathbb{Z}^*_{D_{com}}}} \frac{|\{(a_0, b_0) | (k + 2(a_0 + mb_0)) \in \mathbb{Z}^*_{D_{com}}\}|}{|S|} \\ &> \frac{1}{D_{com}} \cdot \sum_{\substack{\Delta \equiv k \pmod{D_{com}} \\ k \in \mathbb{Z}^*_{D_{com}}}} \frac{\varphi(D_{com})^2}{D^2_{com}} = (\frac{\varphi(D_{com})}{D_{com}})^3 \end{aligned}$$

Therefore, the following inequality holds:

$$(\frac{\varphi(D_{com})}{D_{com}})^3 < P(\gcd((nrd(g^f), D_{com})) = 1)$$

□

**Theorem 2 (Estimation).** *With the same notations in Lemma 1 and Lemma 2, if we assume $g_1$ is uniformly random in $\mathbb{Z}_{Dcom}$, an estimate for the probability that $nrd(g^f)$ and $D_{com}$ are coprime can be expressed as:*

$$P(\gcd(nrd(g^f), D_{com}) = 1) \approx (\frac{\varphi(D_{com})}{D_{com}})^2$$

.

*Proof.* Since $g_1$ is uniformly random in $\mathbb{Z}_{Dcom}$, we can approximate as follows:

$$P(\gcd(k+2g_1, D_{com}) = 1) \approx \frac{\varphi(D_{com})}{D_{com}},$$

where $\varDelta \equiv k \pmod{D_{com}}$. As a result,

$$P(\gcd(\varDelta \cdot (\varDelta + 2g_1), D_{com}) = 1)$$

$$= \frac{1}{D_{com}} \cdot \sum_{\substack{\varDelta \equiv k \pmod{D_{com}} \\ k \in \mathbb{Z}^*_{D_{com}}}} P(gcd(k+2g_1, D_{com}) = 1) \approx (\frac{\varphi(D_{com})}{D_{com}})^2$$

□

**Table 2.** The probability of generating order $\mathcal{O}_0$ through fault injection.

| NIST level | Upper bound $\frac{\varphi(D_{com})}{D_{com}}$ | Lower bound $(\frac{\varphi(D_{com})}{D_{com}})^3$ | Approximation $(\frac{\varphi(D_{com})}{D_{com}})^2$ | Simulation $(\times 100{,}000)$ |
|---|---|---|---|---|
| **NIST-I** | 0.59843 | 0.21430 | 0.35811 | 0.35782 |
| **NIST-III** | 0.53314 | 0.15154 | 0.28424 | 0.27910 |
| **NIST-V** | 0.52081 | 0.14126 | 0.27124 | 0.26797 |

Table 2 shows the theoretical upper bound, lower bound, approximation of the probability that the faulty ideal $I^f_{com}$ generated by the above fault injection becomes $\mathcal{O}_0$ for each NIST security level parameter of SQIsign, and the statistical probability values when simulated 100,000 times with the actual reference code. We can see that the probabilities are all significant for NIST-I,III,V.

## 3.2 Loop-aborts while computing isogeny

As mentioned in Section 2.1, SQIsign uses the Montgomery curve, and consequently, during the computation of the commitment isogeny $\phi$, it separates the operations into two kernels: $E[p+1] \cap ker(\phi)$ and $E[p-1] \cap ker(\phi)$. Thus, in the implementation of the commitment isogeny, the process is divided into two parts, one for computing the isogeny with a kernel corresponding to $E[p+1] \cap ker(\phi)$ (refer to lines 4 to 5 of Algorithm 2), and the other for computing the isogeny with a kernel corresponding to $E[p-1] \cap ker(\phi)$ (refer to lines 13 to 20 of Algorithm 2).

The operations for each kernel involve two nested for-loops. The outer loop sequentially selects distinct prime factors from a table, and the inner loop computes the isogeny for each prime factor's power. The factors and their powers are pre-stored in a table. (Therefore we assume the attacker knows the table's order.)

However, a notable issue arises in the outer for-loop concerning the index value $i$ used for the $E[p+1] \cap ker(\phi)$ kernel. The index $i$ is not reset after the

**Algorithm 3** Odd-degree isogeny computation and the basis evaluation in lines 4 to 5 of Algorithm 1

```
Input: A domain curve $E$
Input: A generator $K^{\pm}$ of $E[p \pm 1] \cap ker(\phi)$, respectively
Input: A basis $(P, Q)$ of $E[D]$ such that $gcd(D, deg(\phi)) = 1$
Output: An image curve $E_c$
Output: An evaluated basis $(\phi(P), \phi(Q))$
1: Let $deg(\phi) = p_1^{e_1} \cdot p_2^{e_2} \cdots p_n^{e_n}$ for $n \in \mathbb{Z}^+$ such that $p_i|(p+1)$ for $1 \leq i \leq h$ and
   $p_i|(p-1)$ for $h < i \leq n$.
2: $E_{1,0} := E$
3: $P_{1,0}, Q_{1,0} := P, Q$
   Evaluate isogenies with kernel in $E[p+1]$
4: for $i \in \{1, ..., h\}$ do
5:     $K_i^+ := [p_{i+1}^{e_{i+1}} \cdot p_{i+2}^{e_{i+2}} \cdots p_h^{e_h}]K^+$
6:     for $j \in \{1, ..., e_i\}$ do
7:         $K_{i,j}^+ := [p_i^{e_i - j}]K_i^+$
8:         Compute isogeny $\phi_{i,j}$ of $deg(\phi_{i,j}) = p_i$ corresponding to the kernel $K_{i,j}^+$
9:         $E_{i,j} := \phi_{i,j}(E_{i,j-1})$, $K^+ := \phi_{i,j}(K^+)$, $K^- := \phi_{i,j}(K^-)$
10:        $P_{i,j}, Q_{i,j} := \phi_{i,j}(P_{i,j-1}), \phi_{i,j}(Q_{i,j-1})$
11:    end for
12: end for
   Evaluate isogenies with kernel in $E[p-1]$
13: for $i \in \{h+1, ..., n\}$ do
14:    $K_i^- := [p_{i+1}^{e_{i+1}} \cdot p_{i+2}^{e_{i+2}} \cdots p_n^{e_n}]K^-$
15:    for $j \in \{1, ..., e_i\}$ do
16:        $K_{i,j}^- := [p_i^{e_i - j}]K_i^-$
17:        Compute isogeny $\phi_{i,j}$ of $deg(\phi_{i,j}) = p_i$ corresponding to the kernel $K_{i,j}^-$
18:        $E_{i,j} := \phi_{i,j}(E_{i,j-1})$, $K^- := \phi_{i,j}(K^-)$
19:        $P_{i,j}, Q_{i,j} := \phi_{i,j}(P_{i,j-1}), \phi_{i,j}(Q_{i,j-1})$
20:    end for
21: end for
22: $E_c := E_{n,e^n}$
23: $\phi(P), \phi(Q) := P_{n,e^n}, Q_{n,e^n}$
24: return $E_c, (\phi(P), \phi(Q))$
```

first for-loop; instead, it continues to be used in the second for-loop. In other words, if the first for-loop concludes with $i = h$, the second for-loop begins with $i = h + 1$. From an attacker's perspective, this operation can be seen as a single for-loop, exhibiting a loop-aborts vulnerability in the isogeny computation process proposed in [15].

An attacker can inject faults at the point where index $i$ is computed, randomizing $i$ and inducing loop-aborts at the desired moment. For example, injecting a fault immediately after the computation of $i = h'$ would prevent $i = h' + 1$ from proceeding, leading to termination of the for-loop and outputting an isogeny of degree $p_1^{e_1} \cdot p_2^{e_2} \cdot \dots \cdot p_{h'}^{e_{h'}}$.

## 4 Fault attack on SQIsign

The vulnerabilities discussed in the previous section can be exploited to inject faults into the commitment phase of SQIsign to reduce the degree of commitment isogeny or to make the commitment ideal $\mathcal{O}_0$, i.e., the norm is 1. If the attacker can recover the isogeny between the starting curve $E_0$ and the public key curve $E_A$ from the faulty signature and public information, the attacker can use the isogeny as a secret key. In this section, we introduce the attacker model for deterministic SQIsign and randomized SQIsign, and present a scenario of key recovery fault attack in each situation, i.e., recovering the isogeny between $E_A$ and $E_0$, and analyze the time complexity and number of queries required for the attack.

### 4.1 Key recovery fault attack for deterministic SQIsign

The key recovery fault attack on deterministic SQIsign is performed under the following assumptions. First, the attacker is allowed to make multiple queries to SQIsign signing oracle using the same key and messages. Second, the oracle generates a signature for each query and the attacker receives these signatures. Third, the attack assumes a first-order fault situation, which allows the attacker to inject a fault once during the oracle's operation.

To illustrate the attack scenario, we explore the data flow of the SQIsign signing protocol when a first-order fault is injected as described in Section 3.2. Initially, a faulty commitment isogeny $\phi^f_{com}$ is generated due to the fault injection in the commitment phase. This affects the computation of the basis points $P_{0,chall}$ and $Q_{0,chall}$ for $E_0[D_{chall}]$, yieding $P^f_{1,chall}$ and $Q^f_{1,chall}$. Consequently, a faulty curve $E^f_1$ and a faulty kernel $K^f_{chall}$ are generated, leading to the computation of the faulty challenge isogeny $\phi^f_{chall}$. The faulty challenge isogeny leads to the generation of $I^f_{chall}$ after decomposing $K_{chall}$ along the basis $P^f_{1,chall}$ and $Q^f_{1,chall}$. The oracle continues to generate $r^f$, $s^f_1, s^f_2$, components of the signature, using $E^f_1$ and $E^f_2$. Detailed steps are omitted as $r^f$, $s^f_1, s^f_2$ are not utilized in the attack. Refer to Algorithm 4 for detailed steps. In the response phase, the oracle computes $\overline{I_{secret}} \cdot I_{com} \cdot I^f_{chall}$ to obtain $J^f_{res}$ and uses *signingKLPT* to generate

**Algorithm 4** First-order fault injected SQISign signing protocol (simplified)

```
Input: A Message $M$
Input: A secret key $sk$
Output: A faulty signature $(\sigma^f, r^f, s_1^f, s_2^f)$
    Commitment process
 1: Sample $a_0, b_0$ randomly from interval $[1, D_{com}]$ until $gcd(a_0, b_0, D_{com}) = 1$.
 2: $I_{com} := \mathcal{O}_0 < \gamma(a_0 + b_0\bar{\theta}), D_{com} >$
 3: $K_{com} := a_0 P_0 + b_0 \theta(P_0)$, where $P_0 \in B_{0,D_{com}}$
 4: Compute a faulty isogeny $\phi_{com}^f : E_0 \to E_1^f$.
 5: $P_{1,chall}^f := \phi_{com}^f(P_{0,chall}), Q_{1,chall}^f := \phi_{com}^f(P_{0,chall})$
    Challenge process
 6: $a_1^f, b_1^f := H(M, E_1^f)$
 7: Find a $D_{chall}$-torsion basis $(P_{E_1}^f, Q_{E_1}^f)$ on $E_1^f$ deterministically.
 8: $K_{chall}^f := a_1^f P_{E_1}^f + b_1^f Q_{E_1}^f$
 9: Compute a faulty isogeny $\phi_{chall}^f : E_1^f \to E_2^f$ with the kernel $\langle K_{chall}^f \rangle$.
10: Find integers $\alpha^f, \beta^f$ such that $[\alpha^f]P_{1,chall}^f + [\beta^f]Q_{1,chall}^f = K_{chall}^f$.
11: Compute a left $\mathcal{O}_0$-ideal $(I'_{chall})^f$ corresponding to the isogeny with the kernel
    $\langle [\alpha^f]P_{0,chall} + [\beta^f]Q_{0,chall} \rangle$.
12: $I_{chall}^f := [I_{com}]_*(I'_{chall})^f$
13: Given $K_{chall}^f$, deterministically find another $D_{chall}$-torsion point $L_{chall}^f$ such that
    a pair $(K_{chall}^f, L_{chall}^f)$ is the basis of $E_1^f[D_{chall}]$.
14: $(L'_{chall})^f := \phi_{chall}^f(L_{chall}^f)$
15: Find a $D_{chall}$-torsion basis $(P_{E_2}^f, Q_{E_2}^f)$ on $E_2^f$ deterministically.
16: Find integers $s_1^f, s_2^f$ such that $[s_1^f]P_{E_2}^f + [s_2^f]Q_{E_2}^f = (L'_{chall})^f$.
17: Given $((L'_{chall})^f, s_1^f, s_2^f)$, deterministically find another $D_{chall}$-torsion point
    $(M'_{chall})^f$ such that a pair $((L'_{chall})^f, (M'_{chall})^f)$ is the basis of $E_2^f[D_{chall}]$.
18: Compute an isogeny $\widehat{\phi_{chall}^f} : E_2^f \to E_1^f$ with the kernel $\langle (L'_{chall})^f \rangle$.
19: $M_{chall}^f := \widehat{\phi_{chall}^f}((M'_{chall})^f)$
20: Find $r^f$ such that $K_{chall}^f = [r^f]M_{chall}^f$.
    Response process
21: $J_{res}^f := \overline{I_{secret}} \cdot I_{com} \cdot I_{chall}^f$
22: Find an ideal $(J'_{res})^f$ of norm $2^e$ which is equivalent to $J_{res}^f$ such that $(J'_{res})^f \cdot \overline{I_{chall}^f}$
    is cyclic using $SigningKLPT$.
23: Convert the ideal $(J'_{res})^f$ into the isogeny $\sigma^f$ of degree $2^e$ using $sk$.
24: return $(\sigma^f, r^f, s_1^f, s_2^f)$
```

an equivalent ideal $(J'_{res})^f$ with a norm of $2^e$. Finally, the oracle converts the isogeny $\sigma^f$ corresponding to $(J'_{res})^f$ and yields $(\sigma^f, r^f, s_1^f, s_2^f)$ as the signature. Note that the codomain of $\sigma^f$ is not $E_2^f$. Since the domain of $\phi^f_{chall}$ is $E_1^f$, and the left order of $I^f_{chall}$ is $\mathcal{O}_1$ instead of $\mathcal{O}_1^f$, the curve corresponding to the right order of $I^f_{chall}$ is not $E_2^f$. For the attacker, this implies that $E_2^f$ can be obtained from the signature of the fault-injected SQIsign, but neither $E_{1^f}$ nor $E_1$ can be recovered through this.

The key recovery scenario of the first-order fault attack is as follows. The attacker first interacts with the SQIsign signing oracle, querying the same message $n+1$ times. The final query is processed normally to obtain a valid signature, while the preceding $n$ queries are faulted during the oracle's operation, leading to loop aborts as described in Section 3.2 and resulting in faulty signatures. If we denote the loop index in Algorithm 3 corresponding to the fault injection point of the $j$-th query as $i_j$, the faults are injected such that $i_1 > i_2 > ... > i_n$.

The attacker can exploit these faulty signatures to reconstruct the isogeny between $E_0$ and $E_1$. Consider a scenario in which the attacker tries to recover a component of the commitment isogeny from the $j$-th faulty signature. Now we denote by $\psi : (E_1)_{j-1} \to E_1$ the isogeny that the attacker has recovered using $j-1$ faulty signatures, and by $\psi_j : (E_1)_j \to (E_1)_{j-1}$ the isogeny to be recovered at the $j$-th stage. If the degree of the commitment isogeny $\phi_{com}$ is $p_1^{e_1} \cdot p_2^{e_2} \cdot \dots \cdot p_n^{e_n}$, then the degree of $\psi_j$ is given by $\prod_{k=i_j}^{i_j-1} p_k^{e_k}$, denoted as $\widetilde{p_{i_j}}$. We also denote the faulty commitment isogeny $\phi^f_{com}$ from the $j$-th fault-injected SQIsign signing oracle operation as $(\phi_{com})_j$ (refer to line 4 of Algorithm 4). Similarly, $(I'_{chall})^f$ and its corresponding isogeny are represented as $(I'_{chall})_j$ and $(\phi'_{chall})_j$(refer to line 12 of Algorithm 4), respectively. The outcome of $[I_{com}]_*(I'_{com})_j$ is represented as $(I_{chall})_j$, and its corresponding isogeny is denoted as $(\phi_{chall})_j$.

In this context, the commitment isogeny $\phi_{com}$ can be expressed as $\psi \circ \psi_j \circ (\phi_{com})_j$. Given that $D_{com}$ and $D_{chall}$ are coprime, the following holds:

$$[\phi_{com}]_*(\phi'_{chall})_j = [\psi \circ \psi_j \circ (\phi_{com})_j]_*(\phi'_{chall})_j = [\psi]_*[\psi_j]_*[(\phi_{com})_j]_*(\phi'_{chall})_j.$$

The attacker can brute force all isogenies of degree $\widetilde{p_{i_j}}$ whose codomain is the domain of $\psi$, denoted as $\psi_{j,k} : (E_1)_{j,k} \to (E_1)_{j-1}$ for $k \in \{1, \dots, \widetilde{p_{i_j}} + 1\}$ and operate on $[\psi]_*[\psi_{j,k}]_*[(\phi_{com})_j]_*(\phi'_{chall})_j$ for each $\psi_{j,k}$. If $\psi_{j,k}$ is indeed a component of the commitment isogeny, then the $j$-invariant of the codomain of $[\psi]_*[\psi_{j,k}]_*[(\phi_{com})_j]_*(\phi'_{chall})_j$ should coincide with the $j$-invariant of the $j$-th faulty signature $\sigma_j$. Otherwise, it should differ. Repeating this procedure $n$ times allows the isogeny between $E_0$ and $E_1$ to be recovered using a divide-and-conquer strategy, as described in Algorithm 5. The attacker can then compute the isogeny between $E_0$ and $E_A$ and use it as a secret key.

**Analysis of the attack** The number of queries required for the attack, assuming that fault injection always succeeds, is $n+1$. However, in a real fault injection environment, not all fault injections are always successful. Therefore, the success probability of fault injection must be considered. If we denote the

**Algorithm 5** Recovery of $(E_0, E_1)$-isogeny by divide-and-conquer

```
Input: A public key $E_A$
Input: A message $M$
Input: A valid signature $\sigma$, faulty signatures $(\sigma_1, ..., \sigma_n)$
Input: $deg(\phi_{com}) = p_1^{e_1} \cdot p_2^{e_2} \cdots p_n^{e_n}$
Input: Fault-injected indices $i_1, i_2, ..., i_n$
Output: A recovered commitment isogeny $\psi : E_0 \to E_1$
Output: $success$
 1: $i_0 := n$
 2: $(E_1)_0 := E_1$
 3: for $j \in \{1, ..., n\}$ do
 4:     Compute the codomain of $\sigma_j$, denoted as $(E_2)_j$.
 5:     $\widetilde{p_{i_j}} := \prod_{k=i_j}^{i_{j-1}} p_k^{e_k}$
 6:     for $k \in \{1, ..., \widetilde{p_{i_j}} + 1\}$ do
 7:         Compute a candidate isogeny $\psi_{j,k} : (E_1)_{j,k} \to (E_1)_{j-1}$
 8:         $(a_1)_{j,k}, (b_1)_{j,k} := H(M, (E_1)_{j,k})$
 9:         Find a $D_{chall}$-torsion basis $(P_{(E_1)_{j,k}}, Q_{(E_1)_{j,k}})$ on $(E_1)_{j,k}$ deterministically.
10:         $(K_{chall})_{j,k} := (a_1)_{j,k} \cdot P_{(E_1)_{j,k}} + (b_1)_{j,k} \cdot Q_{(E_1)_{j,k}}$
11:         Compute an isogeny $(\phi\_)_{j,k} : (E_1)_{j,k} \to (E\_)_{j,k}$ with the kernel $(K_{chall})_{j,k}$.
12:         $\psi_k := \psi \circ \psi_{j,k}$
13:         $(\phi_{chall})_{j,k} := [\psi_k]_* \phi_{\_,j,k}$
14:         Compute the codomain of $(\phi_{chall})_{j,k}$, denoted as $(E_2)_{j,k}$.
15:         if $j((E_2)_j) = j((E_2)_{j,k})$ then
16:             $\psi_j := \psi_{j,k}$
17:             $\psi := \psi \circ \psi_j$
18:         else
19:             $\psi_j := \perp$
20:         end if
21:     end for
22:     if $\psi_j = \perp$ then
23:         $\psi := \perp$
24:         $sccess := False$
25:         return $\psi, sccess$
26:     end if
27: end for
28: $sccess := True$
29: return $\psi, sccess$
```

probability as $\mu$, the number of queries required for the attack scenario by the attacker can be seen to be $O(\frac{n}{\mu})$.

The time complexity of the attack needs to consider two factors. First, the time complexity of recovering the isogeny through brute force. In the proposed attack scenario, a meet-in-the-middle attack cannot be performed during isogeny recovery, so the brute-force time complexity of each isogeny can be seen to be $O(\widetilde{p_{i_j}})$. Second, the number of isogenies that need to be recovered by brute force. This is the same as the number of queries, $n$, by the attacker. If we denote the largest value among $\widetilde{p_{i_j}}$ where $j \in \{1, \cdots, n\}$ as $\widetilde{p_{max}}$, then considering both time complexities together, the overall time complexity of the proposed attack can be seen to be $O(\widetilde{p_{max}} \cdot n)$.

## 4.2 Key recovery fault attack for randomized SQIsign

The fault attack on the randomized SQIsign for key recovery is performed under different assumptions than those of the deterministic one. Firstly, the attacker is allowed to query the SQIsign signing oracle multiple times using their messages. Note that this attack does not require the oracle to operate with the same key, as the attacker can recover the secret key with just one successful fault injection. Secondly, the oracle generates a signature for each query, which are received by the attacker. Lastly, the attack assumes a second-order fault scenario, allowing the attacker to inject a fault twice during the oracle's operation.

We now examine the data flow of the SQIsign signing protocol when two faults are injected as detailed in Sections 3.1 and 3.2. The primary difference from the first-order fault-injected SQIsign signing protocol is that in the commitment phase, an additional fault injection creates a faulty commitment ideal $I^f_{com}$, in addition to $\phi^f_{com}$. This influences the computation of $[I^f_{com}]_*(I'_{chall})^f$, yielding a different $I^f_{chall}$ from the first-order scenario in line 12 of Algorithm 6. In the response phase, the ideal operation $\overline{I_{secret}} \cdot I^f_{com} \cdot I^f_{chall}$ computes $J^f_{res}$, which affects $\sigma^f$. However, as mentioned in Section 3.1, we only consider when $I^f_{com} = \mathcal{O}_0$, so $I^f_{chall}$ becomes $(I'_{chall})^f$ and $J^f_{res}$ becomes $\overline{I_{secret}} \cdot \mathcal{O}_0 \cdot I^f_{chall} = \overline{I_{secret}} \cdot I^f_{chall}$.

The key recovery scenario of the second-order fault attack is as follows. The attacker first interacts with the SQIsign signing oracle, querying one time. The oracle are faulted during operation, leading to randomization as described in Section 3.1 and loop abort as described in Section 3.2, resulting in a faulty signature. The attacker can exploit the faulty signature to reconstruct the isogeny between $E_0$ and $E_{\sigma^f}$, the codomain of $\sigma^f$.

We now denote the loop index in Algorithm 3 corresponding to the fault injection point as $i$, and the isogeny that the attacker aims to recover using the faulty signature as $\psi : (E_0) \to E_{\sigma^f}$. If the degree of the commitment isogeny $\phi_{com}$ is $p_1^{e_1} \cdot p_2^{e_2} \cdot \cdots \cdot p_n^{e_n}$, then the degree of $\psi$ is given by $\prod_{k=1}^{i} p_k^{e_k}$, denoted as $\widetilde{p}$. The isogeny corresponding to $(I'_{chall})^f$ is represented as $(\phi'_{chall})^f$.

Referring to line 11 of Algorithm 6, the kernel generator of $(\phi'_{chall})^f$ is $[\alpha^f]P_{0,chall}+[\beta^f]P_{0,chall}$, hence $(\phi'_{chall})^f = [\phi^f_{com}]^*\phi^f_{chall}$. Since $I^f_{chall} = (I'_{chall})^f$, the isogeny corresponding to $I^f_{chall}$ is $[\phi^f_{com}]^*\phi^f_{chall}$. Meanwhile, the codomain

**Algorithm 6** Second-order fault injected SQISign signing protocol (simplified)

```
Input: A Message $M$
Input: A secret key $sk$
Output: A faulty signature $(\sigma^f, r^f, s_1^f, s_2^f)$
    Commitment process
 1: Sample $a_0, b_0$ randomly from interval $[1, D_{com}]$ until $gcd(a_0, b_0, D_{com}) = 1$.
 2: $I_{com}^f := \mathcal{O}_0 < (\gamma(a_0 + b_0\bar{\theta}))^f, D_{com} >$
 3: $K_{com} := a_0P_0 + b_0\theta(P_0)$, where $P_0 \in B_{0,D_{com}}$
 4: Compute a faulty isogeny $\phi_{com}^f : E_0 \to E_1^f$.
 5: $P_{1,chall}^f := \phi_{com}^f(P_{0,chall}), Q_{1,chall}^f := \phi_{com}^f(P_{0,chall})$
    Challenge process
 6: $a_1^f, b_1^f := H(M, E_1^f)$
 7: Find a $D_{chall}$-torsion basis $(P_{E_1}^f, Q_{E_1}^f)$ on $E_1^f$ deterministically.
 8: $K_{chall}^f := a_1^f P_{E_1}^f + b_1^f Q_{E_1}^f$
 9: Compute a faulty isogeny $\phi_{chall} : E_1^f \to E_2^f$ with the kernel $\langle K_{chall}^f \rangle$.
10: Find integers $\alpha^f, \beta^f$ such that $[\alpha^f]P_{1,chall}^f + [\beta^f]Q_{1,chall}^f = K_{chall}^f$.
11: Compute a left $\mathcal{O}_0$-ideal $(I'_{chall})^f$ corresponding to the isogeny with the kernel
    $\langle [\alpha^f]P_{0,chall} + [\beta^f]Q_{0,chall} \rangle$.
12: $I_{chall}^f := [I_{com}^f]_*(I'_{chall})^f$
13: Given $K_{chall}^f$, deterministically find another $D_{chall}$-torsion point $L_{chall}^f$ such that
    a pair $(K_{chall}^f, L_{chall}^f)$ is the basis of $E_1^f[D_{chall}]$.
14: $(L'_{chall})^f := \phi_{chall}^f(L_{chall}^f)$
15: Find a $D_{chall}$-torsion basis $(P_{E_2}^f, Q_{E_2}^f)$ on $E_2^f$ deterministically.
16: Find integers $s_1^f, s_2^f$ such that $[s_1^f]P_{E_2}^f + [s_2^f]Q_{E_2}^f = (L'_{chall})^f$.
17: Given $((L'_{chall})^f, s_1^f, s_2^f)$, deterministically find another $D_{chall}$-torsion point
    $(M'_{chall})^f$ such that a pair $((L'_{chall})^f, (M'_{chall})^f)$ is the basis of $E_2^f[D_{chall}]$.
18: Compute an isogeny $\widehat{\phi_{chall}^f} : E_2^f \to E_1^f$ with the kernel $\langle (L'_{chall})^f \rangle$.
19: $M_{chall}^f := \widehat{\phi_{chall}^f}((M'_{chall})^f)$
20: Find $r^f$ such that $K_{chall}^f = [r^f]M_{chall}^f$.
    Response process
21: $J_{res}^f := \overline{I_{secret}} \cdot I_{com}^f \cdot I_{chall}^f$
22: Find an ideal $(J'_{res})^f$ of norm $2^e$ which is equivalent to $J_{res}^f$ such that $(J'_{res})^f \cdot \overline{I_{chall}^f}$
    is cyclic using $SigningKLPT$.
23: Convert the ideal $(J'_{res})^f$ into the isogeny $\sigma^f$ of degree $2^e$ using $sk$.
24: return $(\sigma^f, r^f, s_1^f, s_2^f)$
```

**Algorithm 7** Recovery of $(E_0, E_{\sigma^f})$-isogeny

```
Input: A message $M$
Input: A faulty signature $\sigma^f$
Input: $deg(\phi_{com}) = p_1^{e_1} \cdot p_2^{e_2} \cdots p_n^{e_n}$
Input: A fault-injected index $i$
Output: A recovered isogeny $\psi : E_0 \to E_{\sigma^f}$
Output: $success$
 1: Compute the codomain of $\sigma^f$, denoted as $E_{\sigma^f}$.
 2: $\widetilde{p} := \prod_{k=1}^{i} p_k^{e_k}$
 3: for $j \in \{1, ..., \widetilde{p} + 1\}$ do
 4:     Compute a candidate isogeny $\psi_j : E_0 \to (E_1)_j$
 5:     $(a_1)_j, (b_1)_j := H(M, (E_1)_j)$
 6:     Find a $D_{chall}$-torsion basis $(P_{(E_1)_j}, Q_{(E_1)_j})$ on $(E_1)_j$ deterministically.
 7:     $(K_{chall})_j := (a_1)_j \cdot P_{(E_1)_j} + (b_1)_j \cdot Q_{(E_1)_j}$
 8:     Compute an isogeny $(\phi\_)_j : (E_1)_j \to (E\_)_j$ with the kernel $\langle (K_{chall})_j \rangle$.
 9:     $(\phi_{chall})_j := [\psi_j]^* \phi\_{,j}$
10:     Compute the codomain of $(\phi_{chall})_j$, denoted as $(E_2)_j$.
11:     if $j(E_{\sigma^f}) = j((E_2)_j)$ then
12:         $\psi := \psi_j$
13:         $sccess := True$
14:         return $\psi, sccess$
15:     end if
16: end for
17: $\psi := \perp$
18: $sccess := False$
19: return $\psi, sccess$
```

of $\sigma^f$ corresponds to the right order of $J^f_{res}$, which is also the right order of $I^f_{chall}$. Furthermore, the right order of $I^f_{chall}$ corresponds to the codomain of $[\phi^f_{com}]^*\phi^f_{chall}$, hence the $j$-invariant of the codomain of $\sigma^f$ and $[\phi^f_{com}]^*\phi^f_{chall}$ must be identical.

In this context, the attacker can brute force all isogenies of degree $\widetilde{p}$ that have $E_0$ as their domain. These isogenies are denoted as $\psi_j : E_0 \rightarrow (E_1)_j$. If the isogeny $\psi_j$, which is guessed by the attacker, matches the actual computed faulty commitment isogeny $\phi^f_{com}$, then the $j$-invariant of the codomain of $[\psi_j]^*(\phi\_)_j$ will be the same as the $j$-invariant of $E_{\sigma^f}$. If it does not match, then it will be different. Here, $(\phi\_)_j$ is an isogeny of the kernel $\langle (K_{chall})_j \rangle$, which is computed by the codomain $(E_1)_j$ of the guessed isogeny $\psi_j$(refer to lines 4 to 10 of Algorithm 7). Using this, the attacker can recover the isogeny between $E_0$ and $E_{\sigma^f}$ from the given faulty curve. By composing this with $\sigma^f$, they can compute the isogeny between $E_0$ and $E_A$. This computed isogeny can then be used as a secret key.

**Analysis of the attack** Assuming that every attempt at fault injection is successful, the only factor we need to consider is the probability that a successful fault injection leads to the generation of $\mathcal{O}_0$ as $I^f_{com}$. This probability, denoted as $\lambda$, is discussed in Table 2 of Section 3.1. In addition to this, we also need to consider the success rate of fault injection, which we denote as $\mu$. Therefore, the number of queries required for the attack can be approximated as $O(\frac{1}{\lambda \cdot \mu})$.

When considering the time complexity of the attack, the only factor we need to consider is the recovery of the isogeny through brute-force. Therefore, the time complexity is solely determined by the brute-force recovery of the isogeny, which can be approximated as $O(\widetilde{p})$.

## 5 Conclusion and countermeasures

In this paper, we identify and discuss two vulnerabilities in SQIsign that can be exploited by fault attacks. The first vulnerability relates to the computational process of generating an ideal during the commitment phase of SQIsign. If an fault is injected into the generator during the process, it can cause the ideal to be of order $\mathcal{O}_0$, which acts as a multiplication-by-[1] map on $E_0$ by the Deuring correspondence. We have performed an analysis to determine the range and estimation of the probability of this vulnerability occurring. The second vulnerability lies in the isogeny computation in SQIsign, where the entire operation is performed using a single counter during the commitment phase, making it vulnerable to loop abort. Exploiting these vulnerabilities, we outline attack scenarios for cryptographic key recovery in both deterministic and randomized versions of SQIsign. The former scenario assumes a first-order fault attack scenario, while the latter considers a second-order fault attack environment. In addition, we provide analyses of the number of queries and time complexity associated with each attack strategy.

Both vulnerabilities can be countered using intuitive methods. First, for the loop-abort vulnerability related to isogeny operations, it is possible to address it by checking whether the counter of the for-loop has been filled with the correct number of iterations when the for-loop is functioning normally. This was discussed similarly in [14,15]. If the environment allows for high-order fault attacks, checking the counter in parallel using multiple checkers can increase the security against fault attacks. Likewise, a countermeasure against the fault attack on the computation while generating an ideal can be realized by checking the norm of a commitment ideal. In the high-order fault attack situation, it can also be countered by adding additional checkers and ensuring that all checkers have the expected norm of the commitment ideal.

**Acknowledgement.** We thank the anonymous reviewers for their helpful comments on this work. This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korean Government (MSIT) under Grant NRF-2020R1A2C1011769.

## References

1. Adj, G., Chi-Domínguez, J.J., Mateu, V., Rodríguez-Henríquez, F.: Faulty isogenies: a new kind of leakage. arXiv preprint arXiv:2202.04896 (2022)
2. Banegas, G., Krämer, J., Lange, T., Meyer, M., Panny, L., Reijnders, K., Sotáková, J., Trimoska, M.: Disorientation faults in CSIDH. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 310–342. Springer (2023)
3. Beegala, P., Roy, D.B., Ravi, P., Bhasin, S., Chattopadhyay, A., Mukhopadhyay, D.: Efficient loop abort fault attacks on Supersingular Isogeny based Key Exchange (SIKE). In: 2022 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT). pp. 1–6. IEEE (2022)
4. Bernstein, D.J., De Feo, L., Leroux, A., Smith, B.: Faster computation of isogenies of large prime degree. Open Book Series **4**(1), 39–55 (2020)
5. Campos, F., Kannwischer, M.J., Meyer, M., Onuki, H., Stöttinger, M.: Trouble at the CSIDH: protecting CSIDH with dummy-operations against fault injection attacks. In: 2020 Workshop on Fault Detection and Tolerance in Cryptography (FDTC). pp. 57–65. IEEE (2020)
6. Campos, F., Krämer, J., Müller, M.: Safe-error attacks on SIKE and CSIDH. In: International Conference on Security, Privacy, and Applied Cryptography Engineering. pp. 104–125. Springer (2021)
7. Castryck, W., Decru, T.: An efficient key recovery attack on SIDH. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 423–447. Springer (2023)
8. Chavez-Saab, J., Santos, M.C.R., Eriksen, J.K., Hess, B., Kohel, D., Leroux, A., Longa, P., Meyer, M., Panny, L., Patranabis, S., et al.: SQIsign: Algorithm specifications and supporting documentation (2023), https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/sqisign-spec-web.pdf
9. Cheon, J.H., Choe, H., Devevey, J., Güneysu, T., Hong, D., Krausz, M., Land, G., Möller, M., Stehlé, D., Yi, M.: HAETAE: Shorter lattice-based Fiat-Shamir signatures. Cryptology ePrint Archive (2023)

10. De Feo, L., Kohel, D., Leroux, A., Petit, C., Wesolowski, B.: SQIsign: compact post-quantum signatures from quaternions and isogenies. In: Advances in Cryptology–ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7–11, 2020, Proceedings, Part I 26. pp. 64–93. Springer (2020)
11. De Feo, L., Leroux, A., Longa, P., Wesolowski, B.: New algorithms for the Deuring correspondence: towards practical and secure SQIsign signatures. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 659–690. Springer (2023)
12. Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: Crystals-dilithium: A lattice-based digital signature scheme. IACR Transactions on Cryptographic Hardware and Embedded Systems pp. 238–268 (2018)
13. Eisenträger, K., Hallgren, S., Lauter, K., Morrison, T., Petit, C.: Supersingular isogeny graphs and endomorphism rings: reductions and solutions. In: Advances in Cryptology–EUROCRYPT 2018: 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29-May 3, 2018 Proceedings, Part III 37. pp. 329–368. Springer (2018)
14. Espitau, T., Fouque, P.A., Gérard, B., Tibouchi, M.: Loop-abort faults on lattice-based Fiat-Shamir and hash-and-sign signatures. In: Selected Areas in Cryptography–SAC 2016: 23rd International Conference, St. John's, NL, Canada, August 10-12, 2016, Revised Selected Papers 23. pp. 140–158. Springer (2017)
15. Gélin, A., Wesolowski, B.: Loop-abort faults on supersingular isogeny cryptosystems. In: Post-Quantum Cryptography: 8th International Workshop, PQCrypto 2017, Utrecht, The Netherlands, June 26-28, 2017, Proceedings 8. pp. 93–106. Springer (2017)
16. Kohel, D., Lauter, K., Petit, C., Tignol, J.P.: On the quaternion-isogeny path problem. LMS Journal of Computation and Mathematics **17**(A), 418–432 (2014)
17. Maino, L., Martindale, C., Panny, L., Pope, G., Wesolowski, B.: A direct key recovery attack on SIDH. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 448–471. Springer (2023)
18. Robert, D.: Breaking SIDH in polynomial time. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 472–503. Springer (2023)
19. Tasso, É., De Feo, L., El Mrabet, N., Pontié, S.: Resistance of isogeny-based cryptographic implementations to a fault attack. In: Constructive Side-Channel Analysis and Secure Design: 12th International Workshop, COSADE 2021, Lugano, Switzerland, October 25–27, 2021, Proceedings 12. pp. 255–276. Springer (2021)
20. Ti, Y.B.: Fault attack on supersingular isogeny cryptosystems. In: Post-Quantum Cryptography: 8th International Workshop, PQCrypto 2017, Utrecht, The Netherlands, June 26-28, 2017, Proceedings 8. pp. 107–122. Springer (2017)
21. Vélu, J.: Isogénies entre courbes elliptiques. Comptes-Rendus de l'Académie des Sciences **273**, 238–241 (1971)
22. Wesolowski, B.: The supersingular isogeny path and endomorphism ring problems are equivalent. In: 2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS). pp. 1100–1111. IEEE (2022)