Simple Power Analysis Attack on SQIsign

Anisha Mukherjee[1][0009−0002−5565−5744], Maciej Czuprynko[1][0009−0007−2433−5285], David Jacquemin[1][0009−0001−8386−006X], Péter Kutas[2,3][0000−0002−2043−9542], and Sujoy Sinha Roy[1][0000−0002−9805−5389]

[1] ISEC, Graz University of Technology, Austria
[2] Eötvös Loránd University, Hungary
[3] University of Birmingham, England
{anisha.mukherjee,maciej.czuprynko,sujoy.sinharoy}@tugraz.at
david.jacquemin@student.tugraz.at
kutasp@gmail.com

Abstract. The isogeny-based post-quantum digital signature algorithm SQIsign offers the most compact key and signature sizes among all candidates in the ongoing NIST call for additional post-quantum signature algorithms. To the best of our knowledge, we present the first Simple Power Analysis (SPA) side-channel attack on SQIsign, demonstrating its feasibility for key recovery. Our attack specifically targets secret-dependent computations within Cornacchia's algorithm, a fundamental component of SQIsign's quaternion module. At the core of this algorithm, a secret-derived yet ephemeral exponent is used in a modular exponentiation subroutine. By performing SPA on the modular exponentiation, we successfully recover this ephemeral exponent. We then develop a method to show how this leaked exponent can be exploited to ultimately reconstruct the secret signing key of SQIsign. Our findings emphasize the critical need for side-channel-resistant implementations of SQIsign, highlighting previously unexplored vulnerabilities in its design.

Keywords: Isogeny · SQIsign · Side-Channel Analysis · PQC

1 Introduction

Due to rapid advancements in the field of quantum computing in recent times, cybersecurity experts recommend to shift from classical cryptographic primitives to post-quantum alternatives. After the initial standardization effort for post-quantum Key Encapsulation Mechanisms (KEMs) and digital signatures, the National Institute of Standards and Technology (NIST) selected three digital signature schemes: Dilithium [15], Falcon [20], and SPHINCS+ [4]. Since two (Dilithium and Falcon) of these schemes rely on post-quantum lattice-based hardness assumptions, recognizing the need for greater algorithmic diversity, NIST announced a separate call for "additional digital signature" proposals in 2023 [34]. Interestingly, SQIsign is the only isogeny-based signature scheme that has been submitted to this call.

Isogeny-based cryptographic schemes date back to Couveignes's seminal paper [11] which was then rediscovered by Rostovtsev and Stolbunov [38]. Since then, quite a few different approaches have been used for designing isogeny-based signatures, including those that remain unaffected by recent SIDH attacks [5]. One such approach for constructing signatures from isogenies was proposed by Galbraith, Petit and Silva [21](GPS) that relies on the quaternion version of isogeny path-finding problem, referred to as the KLPT (Kohel, Lauter, Petit, Tignol) algorithm [28]. Proposed in 2020, SQIsign [13] improves upon GPS scheme by providing generalized version of the KLPT algorithm and introducing several other novel algorithms resulting in a practically feasible scheme with a much better performance. In fact, SQIsign stands out among all its isogeny-based predecessors and the rest of its NIST counterparts because it is the most compact post-quantum signature to date. Since the introduction of the first version in 2020, several follow-up variants [4] of the original scheme were proposed such as [18] and the higher-dimensional isogeny variants–SQIsignHD [12], SQIsign2D-West [2], SQIsign2D-East [33] and SQIPrime [16].

Along with theoretical security of the submitted schemes, NIST has broadened its assessment criteria [35] to also include implementation aspects such as performance, cryptanalysis and resistance to side-channel attacks (SCA). Since SQIsign relies on a combination of supersingular elliptic curve and quaternion arithmetic, it presents a broader cryptanalytic surface compared to purely elliptic curve-based schemes. While there are many elliptic curve-based side-channel literature such as [36,19,43], SQIsign is yet to undergo any rigorous cryptanalysis on its quaternion side, except the work by [29] that mounts a fault attack on [13]. Moreover, as stated in the NIST specifications document [6], SQIsign does not include a side-channel-resistant implementation, a critical requirement for practical applications.

Our contributions: In this work, we analyze SQIsign's intricate mathematical framework to uncover side-channel vulnerabilities in its signing procedure. We make the following contributions towards side-channel assisted secret-key recovery of SQIsign.

1. **Identification of a Side-Channel Vulnerability:** As also mentioned by SQIsign in their most recent NIST submission [1, Section 1.1], signature generation in SQIsign follows a complex computational hierarchy involving multiple layers of mathematical function calls. This makes identifying side-channel vulnerabilities in SQIsign substantially non-trivial because, even if the attacker is able to recover an intermediate secret value via side-channels, the complete secret key recovery procedure using the obtained intermediate secret would also follow an intricate and complex back-tracking path.
   In this work, through systematically analyzing this mathematical hierarchy, we identify Cornacchia's algorithm as a key vulnerable spot within SQIsign's

[4] At the time of submission of this manuscript, SQIsign had just successfully moved forward to the second round of the "additional signatures" call with the most recent update to the scheme described is the NIST Round-2 specifications [1].

quaternion module. This algorithm computes a square root with respect to an ephemeral modulus, say m, where m is derived from the secret signing key of SQIsign and is an input to Cornacchia's algorithm. We observe that the existing software implementation of SQIsign leaks m through power side-channels during the square root calculation. By exploiting this leakage, we successfully recover m, exposing a pathway to extract the secret key.

2. **Key Recovery via Backtracking:** We present a systematic framework for reconstructing SQIsign's secret signing key by leveraging side-channel-assisted recovery of the ephemeral input m in Cornacchia's algorithm. By tracing m through the computation path back to the KLPT algorithm, we formally establish its relationship with the secret key. We further prove the feasibility of polynomial-time key recovery through three theorems, demonstrating that once m is extracted, the full signing key can be efficiently reconstructed. We provide details about the key recovery attack for two versions of SQIsign, namely, [13,18] and discuss the possibilities for its extension and limitation with respect to the higher-dimensional variants.
3. **Practical Attack Demonstration:** We experimentally validate our attack using Simple Power Analysis (SPA) on a low-cost ChipWhisperer Nano board. By analyzing power traces of the modular square root computation in SQIsign's software implementation, we successfully recover the ephemeral secret input m to Cornacchia's algorithm. This empirical validation confirms that our attack is practical and capable of signing key extraction.

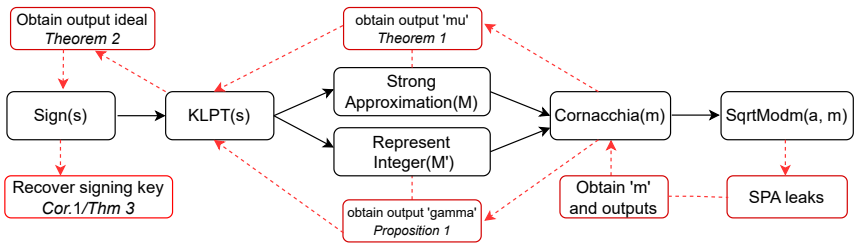


Fig. 1. Attack sketch in red: Steps for recovering the signing key s

Attack sketch: Fig. 1 illustrates the high-level attack procedure, which consists of two main phases. In the first phase, the attacker exploits side-channel leakage by analyzing power traces from the signing procedure. By tracing through multiple levels of function calls down to the modular square root step, the attacker extracts the ephemeral secret-dependent modulus m using simple power analysis techniques. In the second phase, the attacker reconstructs the secret signing key through a purely computational process. By backtracking through the function call hierarchy, the attacker incrementally recovers intermediate secrets at each stage, ultimately deriving the full signing key.

In particular, our investigation begins with the signing process of SQIsign, where a significant portion of the computation involves processing the secret key through quaternion arithmetic enabled by the Deuring correspondence. The KLPT algorithm [28] (described in detail in Sec. 2.2) plays a central role in the signing procedure [13]. We recursively analyze its internal subroutines and identify Cornacchia's algorithm as a prime target for side-channel attacks. This algorithm is used to find a pair of integers (x, y) satisfying the equation $x^2 + d \cdot y^2 = m$ for co-prime integers d and m, where $d = 1$ in our case, and m is derived from the secret key s.

If the input m to Cornacchia's algorithm can be guessed correctly, it becomes possible to reconstruct the secret signing key efficiently. Having established a key recovery strategy under these conditions, we identify specific side-channel leakage points within Cornacchia's execution. A key insight of our work is that, deep down, Cornacchia's algorithm relies on a modular square root computation implemented via the *square-and-multiply* exponentiation method. Taking inspiration from side-channel attacks targeting modular exponentiation in RSA [31,10,23,27,24], we perform experimental Simple Power Analysis (SPA) on the exponentiation used inside Cornacchia's algorithm and successfully recover the exponent, which is a linear function of m. Thereafter, we perform backtracking, as shown in Fig. 1 to reconstruct the signing key of SQIsign.

Organization: The paper is organized as follows. In Sec. 2, we give all the important background information necessary for our paper. Sec. 3 presents a framework which can lead to a polynomial-time key recovery. We show that if an attacker is able to guess the input variables to Cornacchia, then it can successfully recover the signing key in polynomial time. We demonstrate an SPA-style attack on the modular exponentiation algorithm in Sec. 4.

2 Preliminaries

In this section we cover certain mathematical and algorithmic topics necessary to present our results. We use the notations $\mathbb{F}_q$, $\mathbb{Z}$ and $\mathbb{Q}$ to denote a finite field of order q, the set of integers and field of rational numbers respectively. Multiplication between integers, reals or quaternions is represented using the '$\cdot$' operator, we omit the explicit use of '$\cdot$' either to maintain readability or in contexts where this multiplication is obvious.

2.1 Supersingular elliptic curves and quaternion algebras

Let E_1 and E_2 be elliptic curves defined over some finite field $\mathbb{F}_q$. An isogeny is a non-zero rational map that sends the point of infinity of E_1 to the point of infinity of E_2. Being a finite map of curves, it induces a finite field extension on the function fields of the elliptic curves. The degree of the isogeny is defined as the degree of the field extension. An endomorphism of E is an isogeny from E to itself. Endomorphisms, including the zero map, form a ring under addition

and composition. If the endomorphism ring of E is commutative, then we call E *ordinary*, otherwise E is *supersingular*. In this paper, we will only be looking at supersingular elliptic curves. Supersingular elliptic curves can all be defined over $\mathbb{F}_{p^2}$. Detailed discussion is available in [40].

A rational quaternion algebra H is a central simple algebra of dimension four over $\mathbb{Q}$. It always has a presentation $i^2 = a, j^2 = b, ij = -ji$ where $1, i, j, k = ij$ constitute a $\mathbb{Q}$-basis and $a, b \in \mathbb{Q}^*$: $H(a, b) = \mathbb{Q} + i\mathbb{Q} + j\mathbb{Q} + k\mathbb{Q}$. An order in a quaternion algebra is a subring that contains 1 and a $\mathbb{Q}$-basis of the algebra. An order is called maximal if it is maximal with respect to inclusion. To denote a quaternion algebra ramified at p and infinity, we use the notation, $B_{p,\infty}$. In the special case where $p \equiv 3 \pmod 4$ this just amounts to $i^2 = -1$ and $j^2 = -p$. The endomorphism ring of a supersingular curve over $\mathbb{F}_{p^2}$ is a maximal order in $B_{p,\infty}$. Furthermore, every maximal order in $B_{p,\infty}$ is the endomorphism ring of a supersingular elliptic curve. For more details, readers may refer to [42]. SQIsign [6] chooses primes p of size 254, 382 and 510 bits for NIST security levels I,III and V, respectively.

Connecting the world of supersingular elliptic curves to quaternions: The Deuring correspondence [14] is a correspondence between isomorphism classes of supersingular elliptic curves over $\mathbb{F}_{p^2}$ and isomorphism classes of maximal orders in $B_{p,\infty}$. The Deuring correspondence is two-to-one in most cases and one-to-one if and only if the curve can be defined over $\mathbb{F}_p$. We now define the quaternion analogue of an elliptic curve isogeny under this correspondence. The analogue of an isogeny with domain E is a left ideal I of $\mathrm{End}(E)$. The norm of the ideal is defined as the greatest common divisors of all the norms in I and is equal to the degree of the associated isogeny. A natural question is how do we get the "codomain" for a given left ideal I. This is provided by the right order $\mathcal{O}_r(I) = \{\beta \in B_{p,\infty} | I\beta \subseteq I\}$. A connecting ideal between $\mathcal{O}_1$ and $\mathcal{O}_2$ is a left ideal of $\mathcal{O}_1$ whose right order is isomorphic to $\mathcal{O}_2$. An Eichler order is the intersection of two maximal orders. It encodes a particular connecting ideal as $\mathcal{O}_1 \cap \mathcal{O}_2 = \mathbb{Z} + I$ where I is a left ideal of $\mathcal{O}_1$ and a right ideal of $\mathcal{O}_2$. $\mathcal{O}_1 \cap \mathcal{O}_2$ is a $\mathbb{Z}$ sub-lattice of $\mathcal{O}_1$ and $\mathcal{O}_2$ of the same index (sometimes called the level) equal to the norm of the ideal I.

Remark 1. For the following three sections (2.2, 2.3, 2.4) of the necessary technical background, we adopt the following hierarchy for description:

- We follow a left-to-right approach with respect to Fig. 1 and start with the description of the KLPT algorithm in Sec. 2.2. We provide a high-level idea of how it calls the two sub-algorithms RepresentInteger and StrongApproximation during computations. Note that, the description of these algorithms are necessary for the key recovery framework of Sec. 3.
- Next, we move further right in Fig. 1 to describe Cornacchia's algorithm in Sec. 2.3. We defer the description of the specific lower-level constituent functions within Cornacchia until Sec. 3. The description of these algorithms are important to understand the side-channel analysis of Sec. 4. Fig. 3 visual-

izes the significance of these algorithms in the key recovery and side-channel attack.
- Finally, we provide a high-level sketch of the sigma protocol used by SQIsign. The secret signing key is the endomorphism ring of the public curve, E. The KLPT algorithm requires this secret endomorphism ring during a signature generation to compute the response isogenies, as we describe in Sec. 2.4.

We also highlight in red all those steps within an algorithm that involve the secret and are relevant for our attack.

2.2 The KLPT algorithm

Isogeny-based cryptography relies on the mathematical problem of finding isogenies of very large smooth degrees d (usually, $d = l^k$ for a small prime l such as 2 or 3) between supersingular elliptic curves, which is presumed to be computationally infeasible even for a quantum computer. Interestingly, due to the Deuring correspondence, there is a natural quaternion analogue of this problem. Namely given two maximal orders, find a connecting ideal of norm l^e. Such a connecting ideal should exist for a large enough e. Although it is the exact analogue of a hard problem, it admits a polynomial time algorithm, discovered by Kohel, Lauter, Petit and Tignol (KLPT) [28, Section 4], discussed briefly here.

For simplicity we will assume that $p \equiv 3 \pmod 4$ (the general case is not much harder). The first observation is that if one wants to find a path between two maximal orders, then it is enough to connect both maximal orders to one specific order and concatenate the paths. This special order, denoted by $\mathcal{O}_0$ is the endomorphism ring of the elliptic curve $y^2 = x^3 + x$ and contains $1, i, j$ (it is actually generated as a $\mathbb{Z}$-module by $1, i, (1+j)/2$ and $(i+ij)/2$). From now on, we assume that we are looking for a connecting ideal between $\mathcal{O}_0$ and some other maximal order $\mathcal{O}$. One finds a connecting ideal I_0 using an algorithm of Kirschmer and Voight [26]. Naturally, the norm of this ideal is not likely to be of the form l^e, so our goal is to find an equivalent left ideal of $\mathcal{O}_0$ such that its norm is l^e. This is basically equivalent to finding some element in I_0 whose norm is $n(I_0) \cdot l^e$. The next important intermediate step is that one finds an equivalent ideal of prime norm N. This is easy as one just scans through elements of I_0 and since primes are relatively dense, one will find a suitable element very efficiently. This provides with an equivalent ideal I of prime norm N. So our goal is to find $\beta \in I$ such that $n(\beta) = Nl^e$.

KLPT in [13, Alg.9, Step-2] proceeds in two steps. First using RepresentInteger (Alg. 1) it finds an element γ in $\mathcal{O}_0$ of norm Nl^{e_0}. Then it finds an element $\mu_0 \in j\mathbb{Z}[i]$ such that $\gamma(\lambda\mu_0 + N\mu_1)$ is in I for any $\lambda \in \mathbb{Z}$ and $\mu_1 \in O_0$. The final step, called StrongApproximation (Alg. 2) is finding λ, μ_1 such that the norm of $\lambda\mu_0 + N\mu_1$ is l^{e_1}. Now it is clear that the norm of $\gamma(\lambda\mu_0 + N\mu_1)$ is Nl^e (with $e = e_0 + e_1$) and this is what we were looking for.

Require: $M \in \mathbb{Z}$ such that $M > p$. ▷ p is a fixed SQIsign prime
Ensure: $\gamma = x + yi + zj + tij \in \mathcal{O}_0$ with $n(\gamma) = M$.
1: Set $k = \left\lfloor \sqrt{\frac{M}{p \cdot (1+q)}} \right\rfloor$ and sample random integers $z, t \in [\![-k, k]\!]$.
2: Set $M' = M - p \cdot f(z,t)$. ▷ $f(z,t) = z^2 + t^2$
3: **if** Cornacchia$(M') = \perp$ **then**
4: Go back to Step 1.
5: **else**
6: $x, y \leftarrow$ Cornacchia(M'). ▷ Details in Sec. 2.3
7: **end if**
8: $\gamma = (x + iy + j(z + it))$.
9: **return** γ.

Algorithm 1: RepresentInteger$_{\mathcal{O}_0}(M)$ [13]. It calls Cornacchia with the secret input M'. Recovering M' using SPA allows an attacker to guess the output γ by following Proposition 1.

Require: Prime N, l non quadratic residue modulo N, and $C, D \in \mathbb{Z}$.
Ensure: $\mu = \lambda \cdot \mu_0 + N \cdot \mu_1$, with $\mu_0 = j(C + iD)$, $\mu_1 \in \mathcal{O}_0$ such that $n(\mu) = l^{e_1}$ for some $e_1 \in \mathbb{N}$.
1: Select $e_1 \geq p \cdot N^4$ and adjust the parity so that $\frac{l^{e_1}}{p \cdot (C^2 + qD^2)}$ is a quadratic residue mod N. λ is its square root.
2: Select z, t such as $l^{e_1} - p \cdot f(\lambda C + Nz, \lambda D + Nt) = 0 \mod N^2$.
3: Set $M = \frac{l^{e_1} - p \cdot f(\lambda C + Nz, \lambda D + Nt)}{N^2}$.
4: **if** Cornacchia$(M) = \perp$ **then** ▷ Details in Sec. 2.3
5: Go back to Step 2.
6: **else**
7: $x, y \leftarrow$ Cornacchia(M). ▷ Details in Sec. 2.3
8: **end if**
9: **return** $\mu = \lambda j(C + iD) + N(x + iy + j(z + it))$.

Algorithm 2: StrongApproximation$_{l^{e_1}}(N, C, D, \lambda)$ [13]. It calls Cornacchia with the secret input M. Recovering M using SPA allows an attacker to guess the output μ by following Theorem 1.

2.3 Cornacchia's algorithm

Cornacchia's algorithm [9] aims to find a pair of co-prime integers (x, y) as solutions to the quadratic diophantine equation, $x^2 + d \cdot y^2 = m$ where d and m are also co-prime integers. It takes as input m and applies the Euclid's division algorithm to the pair (u_0, m) where, $u_0^2 \equiv -d \pmod{m}$. The call to the Euclid's algorithm breaks when in the successive sequence (r_n) of remainders, it finds an r_k satisfying the relation, $r_k^2 < m \leq r_{k-1}^2$. This translates to employing the so-called 'half-GCD' algorithm in practical implementations. The solution is then, $\left(x = r_k, y = \sqrt{(m - r_k^2)/d}\right)$ if y is an integer, otherwise the process is repeated with another modular square root u_0' until all such roots get exhausted. In general, Cornacchia's algorithm applies to any integer d that is coprime to m, however, in the context of SQIsign [6], we are interested in the case of Cornacchia's algorithm when $d = 1$. We mention this algorithm in Alg. 3. The sub-routine halfgcd (Steps $2 - 5$ in Alg. 3) is just the Euclid's GCD algorithm terminated 'half-way', that is, when the condition of $r_k < \sqrt{m}$ is met. We target the modular square root computation in Step 1 of Cornacchia. Hereafter, we use the representations Cornacchia or Cornacchia(m) (depending on whether the context requires the input to be specified) to specifically refer to Alg. 3 ; otherwise we use the generic term 'Cornacchia's algorithm'.

Require: $m \in \mathbb{Z}$ such that d and m are co-prime in $x^2 + d \cdot y^2 = m$.
Ensure: $x, y \in \mathbb{Z}$ s.t. $x^2 + d \cdot y^2 = m$.
1: Compute $u_0 = \sqrt{-d} \mod m$ and set $i, m_0 = 0, m$. ▷ SqrtModm$(-d, m)$, see alg. 6
2: **while** $m_i > \lfloor \sqrt{m} \rfloor$ **do** ▷ halfgcd algorithm [30]
3: Compute $r_i = m_i \pmod{u_i}$.
4: Set $m_{i+1}, u_{i+1}, i = u_i, r_i, i + 1$.
5: **end while** ▷ End of halfgcd algorithm
6: Set $s_k = m_{i+1}$
7: Check, $\sqrt{(m - s_k^2)/d} \in \mathbb{Z}$.
8: **if** False **then**
9: go to step 1 for a new u.
10: **else**
11: $x = s_k, y = \sqrt{(m - s_k^2)/d}$.
12: **end if**
13: **return** x, y

Algorithm 3: Cornacchia(d, m) [9] requires the computation of the modular square root in Step-1. This computation is targeted via SPA.

Remark 2. In actual implementation [30], the authors use an 'extended' version of the algorithm referred to as ExtendedCornacchia(m_{ext}) whose steps are given in Alg. 4. It ensures that the input m to Cornacchia is almost always a prime, definitely odd and has an easily computable modular square root. This is implemented using the following procedures: a check to ensure that m_{ext} is not

congruent to 3 (mod 4) in Step 3, repeated divisions to remove all powers of primes which are less than 101 and congruent to 1 (mod 4) in Steps 1,4 and finally a probabilistic primality test. We experimentally observed that in many instances, $m_{\text{ext}} = k \cdot m$, with $k = 1$ or a 3 (mod 4) small prime, hence hereafter, for simplicity, we consider $m_{\text{ext}} = m$ for the following parts of the paper and do not distinguish between Cornacchia and its extended version.

Require: $m_{\text{ext}} \in \mathbb{Z}$.
Ensure: $x, y \in \mathbb{Z}$ such that $x^2 + d \cdot y^2 = m$ where m is computed from m_{ext}.
1: $h_1 = 2^{\nu_2} \cdot 5^{\nu_5} \cdot 13^{\nu_{13}} \cdots 101^{\nu_{101}}$ ▷ ν_j denotes the p-adic valuation of m_{ext} w.r.t j
2: $h_3 = 3 \cdot 7 \cdots 83$ ▷ Product of all 3 (mod 4) primes until 101
3: **if** $\gcd(m_{\text{ext}}, h_3) == 1$ **then**
4: Compute $m = \frac{m_{\text{ext}}}{h_1}$.
5: **if** m is a prime **then**
6: Compute $x, y = \mathsf{Cornacchia}(1, m)$. ▷ In SQIsign [6], $d = 1$
7: **end if**
8: **return** x, y
9: **end if**

Algorithm 4: $\mathsf{ExtendedCornacchia}(m_{\text{ext}})$ as given in[30] acts as a pre-cursor to Cornacchia's algorithm.

Remark 3. We recall that Cornacchia's algorithm is called with the specific inputs M and M' whose explicit expressions are provided in Step 3 of StrongApproximation (Alg. 2) and Step 2 of RepresentInteger(Alg. 1) respectively.

2.4 The SQIsign protocol

One way to interpret KLPT is that given two supersingular elliptic curves with known endomorphism rings, one can compute an isogeny between them (a task that is deemed hard otherwise). This motivates the following sigma protocol which is the high-level idea of SQIsign [13]. The public key is a supersingular elliptic curve E and let E_0 be the supersingular elliptic curve, $y^2 = x^3 + x$. Let, $\mathcal{O}_0 = \text{End}(E_0)$ and $\mathcal{O} = \text{End}(E)$ denote the endomorphism rings of E_0 and E, respectively. Here, $\mathcal{O}_0$ and $\mathcal{O}$ are connected by a secret left ideal I_τ of large prime norm corresponding to the secret isogeny τ. The quaternion secret key analogue is the endomorphism ring of E, that is, $\mathcal{O}$. This is visually represented in Fig. 2.

The main steps of the sigma protocol are the following. The prover computes an isogeny, $\phi : E_0 \to E_1$ and sends E_1 to the verifier. The verifier sends a challenge isogeny, $\psi : E_1 \to E_2$ to the prover. The prover responds with an isogeny σ between E and E_2 of degree l^e. The verifier accepts if the response isogeny σ is between E and E_2, has degree l^e, and $\hat{\psi} \circ \sigma$ is cyclic.

Since one can translate endomorphism rings through isogenies [21], the endomorphism ring of E_2 can be computed by the prover. The naive approach here is to use the KLPT algorithm to provide an isogeny between E and E_2. However,

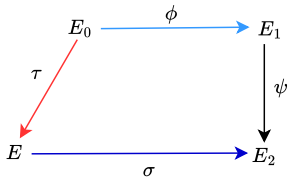


Fig. 2. The sigma protocol in SQIsign [13]. τ represents the secret, ϕ the commitment, ψ the challenge and σ the response isogenies.

the KLPT isogeny has the peculiar feature that it always goes through E_0. This reveals the secret key (i.e., the endomorphism ring of E) after one response. SQIsign uses a modified version of KLPT which connects E and E_2 in a direct fashion, not leaking (conjecturally) any information about the endomorphism ring of E. This can be done by deriving a quaternion analogue of commutative isogeny diagrams. One constructs a left $\mathcal{O}_0$-ideal of the correct norm and pushes it forward via I_τ to a left ideal of $\mathcal{O}$. Of course the difficulty here is to ensure that the right order of this left ideal is isomorphic to $\mathrm{End}(E_2)$. This is ensured by using the Eichler order $\mathcal{O}_0 \cap \mathcal{O}$ and other techniques developed in KLPT.

To be able to respond to the verifier one has to translate the aforementioned ideal into an isogeny. There are two issues that arise in practice. First, the norm of the ideal obtained this way is large, hence its kernel will be defined over a large extension field. This issue is solved by breaking the ideal into smaller chunks, such that the kernel of every chunk is defined over a small extension field.

However, after evaluating the first chunk we need to have an efficient description of the new endomorphism ring (i.e., a way to evaluate every endomorphism of the curve efficiently). Indeed, the explicit Deuring correspondence requires that the kernels of the suitable endomorphisms appear as subgroups of the elliptic curve in order to get the kernels of the corresponding isogenies. This is needed as we have to proceed chunk-by-chunk (using the amount of rational torsion we can utilize). This issue is more complicated, furthermore, there is a difference between the original SQIsign construction [13] and the improved version [18]. We do not recall all the details here but the main concept is that [13] uses KLPT for each of these steps whereas [18] uses a different algorithm called SpecialEichlerNorm given in Alg. 5.

The key idea of SpecialEichlerNorm is that one can give in a sense a simpler representation of the endomorphism rings throughout the isogeny path. This results in a considerable speed-up, mainly because a much better prime p (i.e., relaxed smoothness conditions on $p^2 - 1$) can be chosen this way.

Require: $\mathcal{O}$ a maximal order, I a $(\mathcal{O}_0, \mathcal{O})$-ideal and K a left $\mathcal{O}$-ideal of norm l.
Ensure: $\beta \in \mathcal{O} \backslash (\mathbb{Z} + K)$ of norm dividing T^2.
1: Set $L = \mathsf{RandomEquivalentPrimeIdeal}(I)$, $N = n(L)$ and compute α such that $L = I\alpha$.
2: Compute $K' = \alpha^{-1} \cdot K \cdot \alpha$.
3: Compute $(C : D) = \mathsf{EichlerModConstraint}(L, 1, 1)$.
4: Enumerate all possible solutions of $\mu = \mathsf{FullStrongApproximation}(N, C, D)$ until $\mu \notin \mathbb{Z} + K'$. If it fails, go back to Step 1.
5: **return** $\beta = \alpha \cdot \mu \cdot \alpha^{-1}$

Algorithm 5: $\mathsf{SpecialEichlerNorm}(\mathcal{O}, I, K)$ [18]. It calls a version of $\mathsf{StrongApproximation}$ so the recovery method from Theorem 1 can be extended to this algorithm.

3 The key recovery attack

On the quaternion side, SQIsign represents elements of $B_{p,\infty}$ as a 5-tuple of integers in $\mathbb{Z}^5$ and hence, quaternion arithmetic boils down to linear algebra and integer arithmetic [6, Section 2.4]. We focus on integer arithmetic, especially within Cornacchia's algorithm. As a fundamental building block in SQIsign's quaternion arithmetic [6, Section 2.5.1], Cornacchia's algorithm directly interacts with derived (secret) values related to the secret $\mathcal{O}$. In this section, we first provide details of how we track the path of operations followed by the ephemeral secret, that is, the input to $\mathsf{Cornacchia}$ (see Alg. 3). We investigate the modular square root function in Step 1 of the algorithm that ultimately leads us to the side-channel vulnerable *square-and-multiply* exponentiation algorithm. Then, provided that this input to $\mathsf{Cornacchia}$ is obtained via side-channel observations, we explain the steps that an attacker would back-track in order to recover the signing key.

3.1 Attack localization within Cornacchia

$\mathsf{Cornacchia}$ (Alg. 3) requires the computation of the modular square root of $-d$ in Step 1. This modular square root u_0 is obtained using a modular exponentiation algorithm. In Alg. 6, we give a pseudo-code of the modular square root function as provided in the reference implementation of [30]. We note from Alg. 6 that, **(1)** the core principle of the algorithm relies on the *square-and-multiply* modular exponentiation algorithm, whose pseudo-code is given in Alg. 7. Moreover, **(2)** the different exponents e (highlighted in red) in Alg. 6 are linear functions of m derived based on the residue class of the prime m modulo 4 or 8. The "fast" cases correspond to $m \equiv 3 \pmod 4$ and $m \equiv 5 \pmod 8$ when the exponent is calculated as a linear expression in m. The case when $m \equiv 1 \pmod 8$ is handled using the Shanks-Tonelli algorithm [39,41]. Observations **(1)** and **(2)** together make Alg. 7 a good target for side-channel analysis. The bit-dependent behavior of $\mathsf{Square\&MultiplyExponentiation}(b, e, m)$ introduces a clear execution pattern that can be exploited in power-based side channel analysis that traces

the different power consumption patterns of a squaring and a multiplication operation.

Require: a, m, where m is a prime
Ensure: $\sqrt{a} \pmod m$ if it exists, else the function returns failure
1: $a_{mod} = a \pmod m$
2: **if** $a_{mod} < 0$ **then**
3: $\quad a_{mod} = a_{mod} + m$
4: **end if**
5: **if** $\mathsf{Jacobi}(a_{mod}, m) \neq 1$ **then** ▷ computes the Jacobi symbol, $(\frac{a_{mod}}{m})$
6: $\quad$ **return** FAILURE
7: **end if**
8: **if** $m \equiv 3 \mod 4$ **then** ▷ branch-1 condition w.r.t m
9: $\quad e = (m+1)/4$
10: $\quad \sqrt{a} = a_{mod}^{e} \pmod m$
11: **else if** $m \equiv 5 \pmod 8$ **then** ▷ branch-2 condition w.r.t m
12: $\quad e = (m-1)/4$
13: $\quad temp = a_{mod}^{e} \pmod m$
14: $\quad$ **if** $temp \not\equiv 1 \pmod m$ **then**
15: $\quad\quad e = (m+3)/8$
16: $\quad\quad \sqrt{a} = a_{mod}^{e} \pmod m$
17: $\quad$ **else**
18: $\quad\quad e = (m-5)/8$
19: $\quad\quad temp = (4 \cdot a_{mod})^{e} \pmod m$
20: $\quad\quad \sqrt{a} = (2 \cdot a_{mod} \cdot temp) \pmod m$
21: $\quad$ **end if**
22: **else** ▷ branch-3 condition w.r.t m
23: $\quad \sqrt{a} = \mathsf{ShanksTonelli}(a_{mod}, m)$ ▷ Shanks-Tonelli algorithm if $m \equiv 1 \pmod 8$)
24: **end if**
25: **return** $\sqrt{a} \pmod m$

Algorithm 6: Modular Square Root Computation: $\mathsf{SqrtModm}(a, m)$

3.2 Signing key retrieval from Cornacchia

Let us assume that an attacker has observed side-channel leakage from Cornacchia and has guessed the input m correctly. Now, we show how the attacker can retrieve the outputs of StrongApproximation, RepresentInteger and eventually KLPT in the following part of the section.

Remark 4. In this paper, we consider both the original version [13] and the improved version [18] of SQIsign. Version-1 [6] submitted to NIST is, for our purposes, essentially the same as [18]. Nevertheless changes can be made (e.g., SQISignHD [12] is a somewhat different construction) and we would like to initiate new methods that might be interesting for future applications as well.

Require: Base, exponent and modulus given by $b, e, m \in \mathbb{Z}$ respectively. Also, let, $e = (e_{k-1}, \cdots, e_1, e_0)$ represent the bit decomposition of the exponent e.
Ensure: An integer $a = b^e \pmod m$ in $\mathbb{Z}_m$.
1: Set $a = 1$
2: **for** i from $k-1$ to 0 **do**
3: $\quad a = a \cdot a \pmod m$ ▷ squaring step
4: $\quad$ **if** $e_i = 1$ **then**
5: $\quad\quad a = a \cdot b \pmod m$ ▷ multiplication step
6: $\quad$ **end if**
7: **end for**
8: **return** a

Algorithm 7: Square&MultiplyExponentiation(b, e, m)

Proposition 1. *Suppose we have a way of retrieving the input and hence the outputs of* Cornacchia*. Then we can obtain a small set of valid outputs (amongst which is the actual output) γ of* $\mathsf{RepresentInteger}_{\mathcal{O}_0}$ *in polynomial time if either of the following conditions is satisfied for the reduced norm of γ, $n(\gamma) = M$:*

- *M is of the form Nl^{e_0} where l, e_0 are known and $N < p$.*
- *$M < p\log(p)^c$ for some constant c.*

Proof. If one can obtain M', which is the input of Cornacchia in Step 3 of Alg. 1, then we know x, y along with M' such that $x^2 + y^2 = M'$. Now, we split our algorithm into two cases depending on which condition is satisfied.

If the first condition is satisfied, then we know that $n(\gamma) = M = Nl^{e_0}$ where we know the value of l^{e_0} and $N < p$. We also have, $M = M' - p\cdot(z^2+t^2)$ (Step 2, alg. 1) where we know p and M' but do not know z and t. This implies that we know M modulo p. Now as $M = Nl^{e_0}$ and l^{e_0} is known and coprime to p, we can obtain N modulo p by modular inversion. Since $N < p$, we have obtained the value of N exactly which implies that we have obtained M as well. This in turn implies that the value of $z^2 + t^2$ is known and finding z, t boils down to the sum of two squares theorem which relates the number of solutions to the number of prime factors of $z^2 + t^2$. Since for a random integer N' (here, $N' = (M' - M)/p$) the number of distinct prime factors is typically $\log\log N'$ [22], this means that the solutions of the above equation are bounded by $O(\log{(N')}^c)$ choices where c is a small constant. Hence, we can enumerate over the possible solutions of z, t in polynomial time.

If the second condition is satisfied, then we can proceed in a similar fashion. Here, we just need to guess M in Step 2 of Alg. 1. The condition $M < p\log(p)^c$ implies that the bound k in Step 1 of Alg. 1 is small and thus we only have $O(\log(p)^c)$ choices for z, t.

Theorem 1. *Suppose we have a way of retrieving the input and hence the outputs of* Cornacchia*. Then we can obtain the output of* StrongApproximation *in KLPT if the output size l^{e_1} is known.*

Proof. The goal is to obtain μ by utilizing the input and outputs of Cornacchia. Having access to Cornacchia's input and outputs means we know M and x, y

such that $M = x^2 + y^2$. From Step 3 of Alg. 2 we have the equality,

$$MN^2 = l^{e_1} - p \cdot \big((\lambda C + Nz)^2 + (\lambda D + Nt)^2\big), \tag{1}$$

which simplifies to, $MN^2 \equiv l^{e_1} \pmod{p}$. Since M and l^{e_1} are known, we have two choices for the square root, N modulo p, of N^2. In KLPT, $N \approx \sqrt{p}$ which gives us the exact value of N in the following way: indeed, every residue has two square roots ($\pm a$) modulo p (a and $p - a$). This implies that only one of the square roots of N^2 will be close to $\sqrt{p}$ and the other will be much larger than $p/2$, as ($\sqrt{p} \ll \frac{p}{2}$).

Next, knowing M, N and l^{e_1} in equation 1 implies that we know $(\lambda C + Nz)^2 + (\lambda D + Nt)^2$. From this we can solve the equation of the form, $N' = (\lambda C + Nz)^2 + (\lambda D + Nt)^2 = x_0^2 + y_0^2$ and compute all solutions x_0, y_0. The number of integer solutions to this equation is governed by the sum of two squares theorem, which relates the number of solutions to the number of prime factors of $(\lambda C + Nz)^2 + (\lambda D + Nt)^2$. Since for a random integer N' the number of distinct prime factors is typically $\log \log N$ [22], this means that the solutions of the above equation are bounded by $O(\log (N')^c)$ choices for a small constant c. Hence, by enumerating over all possible solutions, we can find $\lambda C + Nz$ and $\lambda D + Nt$. Since the output of StrongApproximation is $\mu = Nx + Nyi + (\lambda C + Nz)j - (\lambda D + Nt)ij$, we have found μ as we now know every coordinate of it.

Remark 5. In KLPT, l^{e_0} is mostly fixed and l^{e_1} is chosen in a way that StrongApproximation is solvable. For efficiency purposes one would like e_1 as small as possible which would leave only a constant number of choices (but definitely bounded by $\log p$) for e_1. Thus, in the statement of Theorem 1 we made this simplification that l^{e_1} is known which does not affect our attacks (furthermore, as part of the scheme, KLPT can easily be modified to return a fixed e_1).

Theorem 1 and Proposition 1 are the main building blocks that enable key recovery. We will provide two key recovery methods, one on the original SQIsign construction [13] and one on the newer version [18]. First we focus on the original construction.

Theorem 2. *Suppose we have a way of retrieving the input and hence the outputs of* Cornacchia*. Then we can retrieve the output ideal of the KLPT algorithm.*

Proof. The KLPT algorithm, as mentioned before (and described in [28]) has three main steps. The first step is computing γ which is the output of the $\mathsf{RepresentInteger}_{\mathcal{O}_0}$ algorithm. Next comes computing a μ_0 such that $(\mathcal{O}_0\gamma)\mu_0 \equiv I \pmod{N\mathcal{O}_0}$ and then lifting μ_0 to μ whose norm is l^e. Then, $\gamma \cdot \mu$ will be an element of I of norm Nl^e which provides an equivalent ideal $I \sim J$ of norm l^e. Theorem 1 implies that we can obtain μ and Proposition 1 implies that we can obtain γ. Putting these together, we obtain $\gamma \cdot \mu$ which is exactly what we need to get the output of KLPT.

Remark 6. Proposition 1 might not be strictly necessary in certain contexts if the algorithm $\mathsf{RepresentInteger}_{\mathcal{O}_0}$ is performed deterministically as one can just

recompute the algorithm offline. The reason is that Theorem 1 also provides N (not just μ), hence if l^{e_0} is chosen deterministically one does not need to use the Cornacchia oracle again.

Corollary 1. *Suppose we have a way of retrieving the input and hence the outputs of* Cornacchia*. Then we can get the signing key in SQIsign [13] in polynomial time.*

Proof. We focus on [13, Algorithm 9] which provides the ideal-to-isogeny translation. In Step 2, KLPT is used to obtain an equivalent ideal J which is a connecting ideal between $\mathcal{O}_0$ and $\mathcal{O}$ where $\mathcal{O}$ is the endomorphism ring of the public curve. Theorem 2 implies that we can get J which immediately reveals $\mathcal{O}$ as the right order of J is isomorphic to $\mathcal{O}$.

Theorem 2 is not directly applicable to the new SQIsign version [18] as in the ideal-to-isogeny translation algorithm KLPT is no longer used. Instead of computing with entire endomorphism ring, the algorithm SpecialEichlerNorm is called which computes a well chosen endomorphism of the appropriate maximal order. In the next theorem, we show how to apply Theorem 1 to obtain the signing key.

Theorem 3. *Suppose we have a way of retrieving the input and hence outputs of* Cornacchia*. Then we can retrieve the signing key in the improved version of SQIsign [18] in polynomial time.*

Proof. In the improved version of SQIsign [18], we target Step 5 of SpecialEichlerNorm which is a direct call to StrongApproximation algorithm. Theorem 1 implies that we can obtain μ using our Cornacchia approach. Furthermore, the proof of Theorem 1 implies that we also retrieve $n(L) = N$ as $N < p$. Our goal is to compute generators for the ideal L as then the right order of L will be the required secret endomorphism ring $\mathcal{O}$ of the public curve (as the first input of SpecialEichlerNorm is the order $\mathcal{O}$). We outline the method to compute these generators in the following paragraph:

Note that, $\mu \in \mathbb{Z} + L$, so μ alone is not enough to retrieve L. Since $\mu \in \mathcal{O}_0$ one can write it as a 2×2 matrix with entries from $\mathbb{Z}/N\mathbb{Z}$ via the explicit isomorphism $\mathcal{O}_0/N\mathcal{O}_0 \cong M_2(\mathbb{Z}/N\mathbb{Z})$. Elements in L have the property that when they are written as 2×2 matrices, they are not invertible as their norms are divisible by N. Now, $\mu = \lambda + \sigma$, where $\lambda \in \mathbb{Z}$ and $\sigma \in L$ and hence we need to figure out the value of λ. Since $N\mathcal{O}_0$ is contained in L, actually, λ is only determined modulo N. Also, since elements in L correspond to matrices that are not invertible, $\mu - \lambda$ is not an invertible matrix which is equivalent to λ being an eigenvalue of μ. Eigenvalues can be computed efficiently as N is a prime number. In general μ has two eigenvalues which provides two possible choices λ_1 and λ_2 for λ. Elements in L correspond to a set of matrices in $M_2(\mathbb{Z}/N\mathbb{Z})$ whose kernel contains a particular cyclic subgroup of $(\mathbb{Z}/N\mathbb{Z})^2$ (or equivalently a point in $\mathbb{P}^1(\mathbb{Z}/N\mathbb{Z})$). Hence for both λ_i's we compute the kernel of $\mu - \lambda_i$ and compute the corresponding ideal L_i. By computing right orders we get two

candidates $\mathcal{O}_1$ and $\mathcal{O}_2$ for $\mathcal{O}$. Then we compute the supersingular elliptic curves corresponding to these orders and choose the one whose j-invariant matches that of the public curve. The last step can be accomplished in polynomial time and is quite practical using the methods of SQIsign or that of [17].

Summarization of the key recovery framework: An attacker obtains the input to Cornacchia $\xrightarrow{\text{Step-1}}$ finds a candidate output γ of RepresentInteger in Alg. 1 using Proposition 1 $\xrightarrow{\text{Step-2}}$ finds a candidate output μ of StrongApproximation in Alg. 2 using Theorem 1 $\xrightarrow{\text{Step-3}}$ (I) finds a candidate output KLPT using Theorem 2 $\rightarrow$ finds a candidate secret using Corollary 1 for [13] version or, (II) finds a candidate secret using Theorem 3 for [18] version of SQIsign. If the candidate secret does not match the actual secret, the attacker repeats Steps 1 to 3.I/3.II. These steps are shown in Fig. 3.

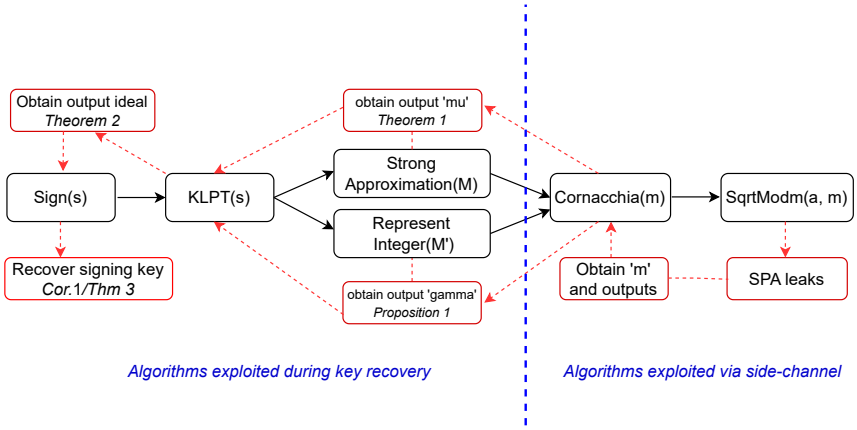


Fig. 3. Strategy for signing key recovery

Remark 7. It would be interesting to attack SigningKLPT [6, Algorithm 17] using our methods as that is common in both versions as the difference lies in the ideal to isogeny translation algorithm. However, the StrongApproximation algorithms used there will reveal the endomorphism ring corresponding to the pushforward of the secret ideal hence it is not immediate how one would get the signing key from that information. We leave this as an open problem for the future.

3.3 New variants of SQIsign

Recently new variants of SQIsign were proposed essentially all leveraging techniques used in the SIDH attack [5]. A way of interpreting the attack is that

one can represent an isogeny by providing evaluations of the isogeny on a sufficiently large subgroup. Verifying whether this representation indeed produces an isogeny between the correct curves of the correct degree can be accomplished by evaluating a chain of higher dimensional isogenies between principally polarized abelian surfaces. This idea was used in SQIsignHD [12] to significantly simplify the signing process of SQIsign but verification in SQIsignHD requires evaluating a chain of $(2, 2, 2, 2)$-isogenies between abelian surfaces which is much slower than verification in the original SQIsign protocol. However, several protocols emerged that modified SQIsignHD in a way that verification only requires 2-dimensional isogenies [16],[2],[33]. We will briefly discuss how our attacks generalize to these 2D variants.

SQIsign2D-East: SQIsign2D-East [33] has the most common algorithms with the original SQIsign protocol as it uses RepresentInteger and StrongApproximation. StrongApproximation is used in the response step as a subroutine of AuxiliaryPath [33, Algorithm 3]. Since it produces a quaternion in $\mathcal{O}_0 \cap \mathcal{O}$ where the level of this Eichler order can be computed as in Theorem 3, one similarly gets the endomorphism ring of the public curve. There is one small subtlety: in Theorem 3 when we invoke StrongApproximation, we know the output norm (as it is a fixed power of l), but, this is not apriori known in SQIsign2D-East. However, an attacker can obtain this value after observing a valid signature (this number actually changes every iteration, unlike in SQIsign). In conclusion, the attack from Theorem 3 can be applied in this context.

SQIsign2D-West: SQIsign2D-West [2] no longer uses StrongApproximation, only RepresentInteger. It is mainly used to evaluate a random isogeny both in the commitment and the response algorithm. However, the random isogeny evaluation algorithm takes an output of the RepresentInteger and then re-randomizes it internally. Due to this internal randomization step, our attacks, at the moment, cannot have a straightforward impact on SQIsign2D-West.

The most recent version of SQIsign [1] submitted to NIST round-2 adopts algorithms from SQIsign2D-West, hence, from a preliminary survey of the documentation, it appears that our key recovery attack is not directly applicable to this version. However, the modular square root algorithm presented in the document [1, Algorithm 3.1] is still essentially the same as presented in Alg. 6 including the modular exponentiation, which leaves an opportunity open for a simple power analysis attack.

SQIPrime: Similar to SQIsign2D-West, SQIPrime [16] also only uses RepresentInteger. However, it doesn't mask using randomization which means our attacks could target the endomorphism ring of the commitment curve. One problem is that SQIPrime uses RepresentInteger on an unknown number, hence, Proposition 1 does not immediately apply. However, the secret integers whose values are required to be known appear in elliptic curve scalar multiplications in the commitment step. Thus, if implemented with a *double-and-add* algorithm,

much like the *square-and-multiply* algorithm, this scalar multiplication could also be targeted as a separate but similar side-channel attack approach as in Sec. 4. Knowing the endomorphism ring of the commitment curve can be exploited in the following way: eventually, after a signature generation, one will have an isogeny representation of a non-smooth isogeny from the public curve to a curve of known endomorphism ring. Then, the attacker can use the quantum polynomial time attack against pSIDH [7] which recovers the secret endomorphism ring in quantum-polynomial time. Note however, that this is a significantly weaker attack compared to the attack presented in the paper as it would require extra information and has quantum polynomial time complexity.

4 Side-channel attack experimentation

In the previous section, we demonstrated that if an attacker can guess the input to Cornacchia, they can reconstruct the secret signing key. The key question now is: How does the attacker determine this input? In this section, we outline our approach of recovering the input of Cornacchia's algorithm by analyzing side-channel leakage from the *square-and-multiply* modular exponentiation, i.e., Alg. 7, used for square-root calculations in Alg. 6. We infer the exponent, which has a linear relationship with Cornacchia's input. We keep in mind Remark 3.

In Sec.3, we describe how the inputs to $\mathsf{Cornacchia}(m)$ are preprocessed using Alg. 4 to ensure that the algorithm can find a solution to the given Diophantine equation. Notably, Step 2 of Alg. 4 significantly reduces the likelihood of m being congruent to $3 \pmod 4$. In practice, we observe that values of m satisfying $m \equiv 1 \pmod 8$ or $m \equiv 5 \pmod 8$ are passed to $\mathsf{Cornacchia}(m)$. Consequently, either branch-2 or branch-3 is invoked in $\mathsf{SqrtModm}(a, m)$, Alg. 6.

- When branch-2 is invoked, the recovered exponent is one of the three forms given in Steps-12, 15 and 18. To validate a guess for the exponent and hence the signing key, the attacker follows the steps in Sec. 3, that is, from Theorem 1 until Corollary 1 or Theorem 3 depending on which variant of SQIsign being targeted. Thus, in the end the attacker computes the corresponding maximal order for each candidate guess and maps it to its associated supersingular elliptic curve representation. Since the j-invariant of the public curve is known, the attacker can verify the correctness of a guessed secret key by matching the computed j-invariant with the j-invariant of the public curve. A successful match confirms that the attacker has recovered the correct secret key.
- When branch-3 is invoked, the Shanks-Tonelli algorithm involves iterative computations depending on the factorization of $m - 1$ and the structure of quadratic residues. Therefore, the exponent used in the exponentiation varies dynamically making it comparatively harder for a straight-forward inference of the exponent from the observed bits in a power trace. Furthermore, since m changes for every signature, each signing operation provides only one opportunity to collect a power trace for a given m.

Consequently, the success probability of the attack is 50% per Cornacchia call. In case of the two independent calls to Cornacchia, once within StrongApproximation 2 and the other within RepresentInteger 1 in the SQIsign version from [13], one out of the four pairs: $\big(m \equiv 5 \pmod 8$ in Alg.2, $m \equiv 1 \pmod 8$ in Alg.1$\big)$, $\big(m \equiv 1 \pmod 8$ in Alg.2, $m \equiv 1 \pmod 8$ in Alg.1$\big)$, $\big(m \equiv 1 \pmod 8$ in Alg.2, $m \equiv 5 \pmod 8$ in Alg.1$\big)$, $\big(m \equiv 5 \pmod 8$ in Alg.2, $m \equiv 5 \pmod 8$ in Alg.1$\big)$ is suitable for key recovery and since primes $5 \pmod 8$ and $1 \pmod 8$ are equally likely, the number of signature generations the attacker has to observe is at least four. In the case of the improved version [18] that calls only StrongApproximation, the number of signature generations for recovering m is two. Note that in case of a call to branch-1 if m is congruent to $3 \pmod 4$, the above-mentioned technique can also be utilized for recovery of m in the same way as branch-2.

4.1 Simple power analysis of modular exponentiation

Simple Power Analysis (SPA) is a widely used technique in side-channel attacks, where an adversary passively monitors a device's power consumption to infer secret information. Unlike more advanced techniques such as Differential Power Analysis (DPA), SPA relies on directly interpreting power traces rather than statistical correlation. The underlying idea in SPA is quite simple: the power consumption of a device changes based on the functions or operations being executed at a point in time; a computationally expensive operation will consume more resources than a comparatively straightforward one. In our attack, we use SPA to observe power consumption patterns during modular square root computations, exploiting secret-dependent variations to recover secret information.

In Sec. 2.3 and specifically in Alg. 7, we observed the bit-dependent behavior of Square&MultiplyExponentiation(b, e, m). At each step of the Algorithm, there is a squaring and when the processed exponent bit is 1, there is an additional multiplication. In SQIsign's implementation [30], the exponentiation algorithm contains specific squaring and multiplication functions along with modular reduction after every squaring and multiplication operation. Implementing these operations differently leads to distinct patterns in the power traces that let us use SPA and retrieve the secret exponent.

To demonstrate feasibility, we executed the attack on a ChipWhisperer Nano board featuring the STM32F030 microcontroller, using its built-in ADC for trace acquisition. The code for *square-and-multiply* exponentiation was compiled with `arm-none-eabi` version 10.2.1 with optimization level `-O0`. Due to the limited ROM and SRAM capacity of the microcontroller, we implemented the exponentiation for the 54 bit toy parameter set from [37]. Since this choice is a practical constraint imposed by hardware limitations, the attack can naturally be extended to actual NIST security level SQIsign parameters following our approach. The only difference will be that the execution time of the entire algorithm, and hence the trace, will be longer. We implemented the naive implementation described in Alg. 7 with dedicated squaring and multiplication functions. As for the reduction modulo m, the Montgomery reduction was used, where the needed constants were precomputed before the start of the exponentiation.

We show part of the result of an exponentiation in Fig. 4. We highlight the visually distinguishable multiplications and squaring operations. Furthermore, we also distinguish the modular reduction which follows after both operations. In particular, we notice that the reduction contains two multiplications itself. In the given figure, we are able to 'read' the first four bits of the exponentiation which are 1001. By continuing this procedure for the whole exponentiation, we are able to retrieve the complete corresponding exponent with 100% success probability. The retrieved exponent would follow either of the branch conditions explained in the previous sub-section.

Indeed, when the prime m is congruent to 5 (mod 8), the attacker can link the retrieved exponent directly to either of the three forms given in Steps-12, 15 and 18 of SqrtModm 6. In the case of m being congruent to 1 (mod 8), the exponent obtained via observing the traces cannot be successfully used to get m. From the recovered exponent, the attacker can iterate over the three choices in Steps-12, 15 and 18 of Alg. 6 one-by-one to compute the corresponding three candidates for m. These three candidates for m are in fact, the three possible choices for the input of Cornacchia. The attacker then repeats from Steps 1 through 3.I/3.II depending on the SQIsign version until a match for the secret key is found based on the verification condition of Corollary 1 (or, Theorem 3). If, even after iterating through all the three possibilities, the attacker does not hit a match for the secret key, then Alg. 6 executed branch-3 and m was congruent to 1 (mod 8). In such a scenario, the attacker has to repeat the process starting from power trace collection until Steps 1-3.I/3.II. In our SageMath implementation[5] of these steps with 54-bit parameter set, the time taken to find matches for γ and μ is in the order of 10^{-3} seconds.

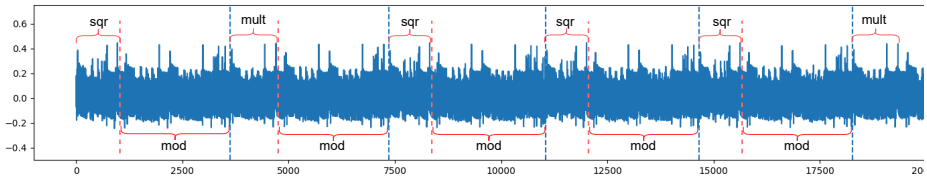


Fig. 4. SPA trace (power vs time) of the exponentiation showing the squaring, multiplication and modular reduction.

4.2 Countermeasures

As [24] discusses, there can be two broad approaches to prevent leakage of sensitive information through physical side-channel attacks such as an SPA. The

[5] `git@github.com:anishamukh/Key-Recovery-of-SQIsign.git`

first approach aims to remove or hide any correlation of the underlying secret-dependent operations from the observed side-channel patterns, while the second removes correlation of the side-channel from the secret data. [24] states that the latter approach is more hardware-oriented and aims to make the device more tamper-resistant. We discuss countermeasures related to the former approach. This approach works either by employing specific algorithmic tricks designed explicitly to make side-channel observations independent of the branch conditions related to the secret value, or introducing randomness to the secret-dependent parameters. Such countermeasures along this approach have been developed to protect the classical *square-and-multiply* exponentiation from SPA attacks.

Algorithmic tweaks for regularity of operations: The usual *square-and-multiply* exponentiation contains conditional branches that either execute a squaring or a squaring followed by a multiplication depending on the exponent bit being 0 or 1. This "irregularity" in the sequence of operations are what allows an SPA attacker to distinguish between specific exponent bits. One intuitive way to alleviate this "irregularity" of operations can be circumvented by replacing the bit-dependent *square-and-multiply* exponentiation with algorithms that perform the same amount of computations irrespective of the secret exponent bit. Two such widely-known algorithms are the *square-and-multiply always* exponentiation algorithm and the Montgomery ladder exponentiation [25,8,32]. The *square-and-multiply always* exponentiation performs both squaring and multiplication operations at every iteration, regardless of the exponent bit's actual value. When the exponent bit is 1, the multiplication step involves the intended operand, and when the exponent bit is 0, the algorithm multiplies by a dummy value. Consequently, each iteration follows a uniform operation sequence thereby producing consistent power consumption patterns that obscure secret-dependent information. The Montgomery ladder follows a similar algorithmic flow as the *square-and-multiply always* exponentiation but without dummy multiplications.

Note that in SQIsign, the ephemeral secret, that is, the input to Cornacchia changes with every signature execution. Thus, an attacker has only one chance at side-channel analysis for a particular value of the ephemeral secret. Using the aforementioned regular algorithms such as the Montgomery ladder, therefore, makes SPA essentially impractical. Moreover, it means that an attacker cannot collect enough traces with respect to that particular ephemeral secret value for mounting advanced attacks such as Differential Power Analysis. In [3], the authors consider the case of a scalar multiplication in ECC (elliptic curve cryptography) where the secret scalar is an ephemeral. They provide a list of the different types of possible passive attacks other than SPA such as horizontal (collision), horizontal correlation and template attacks that can still be exploited to recover the ephemeral scalar. Such alternative attacks could possibly also be explored w.r.t the *square-and-multiply* exponentiation as it is algorithmically similar to an elliptic curve scalar multiplication. Beyond using regular algorithms, [3] also suggests randomizing the secret value to protect against horizontal correlation

attacks. In the following paragraph, we discuss a randomization method that can be applied to the secret exponent.

Exponent randomization: In this technique [27], the exponent is randomized before performing the exponentiation operation. Specifically, the exponent e can be replaced by $e + r \cdot \phi(m)$ for some random integer r and Euler's totient function $\phi(m)$ that ensures correctness of the result modulo m. This randomization prevents an attacker from obtaining reliable leakage patterns since the observed power traces correspond to the randomized exponent instead of the fixed exponent value. A similar randomization technique could also be applied to the exponents of Alg. 6 before the exponentiation step. It would require computing $\phi(m)$ when m is ephemeral secret. Observing the power consumption traces of the exponentiation algorithm would then provide an attacker with the corresponding bit representation of $e+r\cdot\phi(m)$ instead of e directly. Since the attacker only has one chance of collecting the power trace corresponding to the exponent, such a randomization would increase resistance to correlation attacks.

We note that in this paper, we explore one possible side-channel vulnerability in the signing procedure, namely, in the exponentiation operation within Cornacchia's algorithm. Accordingly, the aforementioned countermeasures provide SPA-resistant alternatives only for the *square-and-multiply* exponentiation in Cornacchia's algorithm. However, several other high-level algorithms involved in signing, or other subroutines within Cornacchia's algorithm, particularly those involving branch-dependent computations, still remain susceptible to side-channel attacks. Therefore, a comprehensive constant-time and side-channel-resistant implementation is necessary for all such algorithms, including Cornacchia's algorithm itself, whenever they process secret or secret-derived intermediate values in the signing procedure.

5 Conclusion

In this work, we presented the first Simple Power Analysis attack on SQIsign, demonstrating its feasibility for key recovery. Our attack exploited secret-dependent computations within Cornacchia's algorithm, a crucial component of SQIsign's quaternion module. By targeting the modular exponentiation used in the computation of modular square roots, we successfully extracted an ephemeral integer m, which is derived from the secret signing key.

Building on this leakage, we developed a systematic framework to trace m back through the KLPT algorithm, formally establishing its connection to the secret key. We showed that once m is recovered, the full signing key can be reconstructed efficiently in polynomial time. Our attack was validated experimentally using a low-cost ChipWhisperer Nano board, confirming its practical applicability. Furthermore, we analyzed the attack's implications across different versions of SQIsign, highlighting vulnerabilities in both the original and improved implementations. We also discussed potential generalizations to two higher-dimensional variants of SQIsign and also presented the limitations of our attack in case of the other variants.

To mitigate the observed side-channel vulnerability, we reviewed countermeasures such as constant-time exponentiation techniques (e.g., Montgomery ladder, square-and-multiply always) and exponent blinding, which prevent SPA leakage. Our findings emphasize the need for side-channel-resistant implementations of SQIsign to ensure its security in real-life applications. Future work includes analyzing advanced side-channel attacks, exploring comprehensive side-channel-resistant signing implementation, and evaluating the impact of countermeasures on performance.

6 Acknowledgements

Anisha Mukherjee, Maciej Czuprynko are supported by the Austrian FWF grant PAT6402023, and Sujoy Sinha Roy is partially supported by the State Government of Styria, Austria, through the Department Zukunftsfonds Steiermark. Péter Kutas is supported by the Hungarian Ministry of Innovation and Technology NRDI Office within the framework of the Quantum Information National Laboratory Program and by the grant "EXCELLENCE-151343", the János Bolyai Research Scholarship of the Hungarian Academy of Sciences and partly by EPSRC through grant number EP/V011324/1.

References

1. Aardal, M.A., Adj, G., Aranha, D.F., Basso, A., Andrés Canales Martínez, I., Chavez-Saab, J., Corte-Real Santos, M., Dartois, P., De Feo, L., Duparc, M., , Komada Eriksen, J., Boris Fouotsa, T., Luiz Gazzoni Filho, D., Hess, B., Kohel, D., Leroux, A., Longa, P., Maino, L., Meyer, M., Nakagawa, K., Onuki, H., Panny, L., Patranabis, S., Petit, C., Pope, G., Reijnders, K., Robert, D., Rodríguez-Henríquez, F., Schaeffler, S., Wesolowski, B.: SQISIGN: Algorithm specifications and supporting documentation. In: Proposal to NIST Standardization of Additional Digital Signature Schemes, Round2. National Institute for Standards and Technology (2025)
2. Basso, A., Dartois, P., Feo, L.D., Leroux, A., Maino, L., Pope, G., Robert, D., Wesolowski, B.: SQIsign2D–West: The fast, the small, and the safer. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 339–370. Springer (2024)
3. Batina, L., Chmielewski, L., Haase, B., Samwel, N., Schwabe, P.: Sok: Sca-secure ECC in software - mission impossible? IACR Trans. Cryptogr. Hardw. Embed. Syst. **2023**(1), 557–589 (2023). https://doi.org/10.46586/TCHES.V2023.I1.557-589, https://doi.org/10.46586/tches.v2023.i1.557-589
4. Bernstein, D.J., Hülsing, A., Kölbl, S., Niederhagen, R., Rijneveld, J., Schwabe, P.: The SPHINCS+ signature framework. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. p. 2129–2146. CCS '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3319535.3363229, https://doi.org/10.1145/3319535.3363229
5. Castryck, W., Decru, T.: An efficient key recovery attack on SIDH. In: EUROCRYPT 2023, Lyon, France, April 23-27, 2023, Part V. vol. 14008, pp. 423–447. Springer (2023). https://doi.org/10.1007/978-3-031-30589-4_15

6. Chavez-Saab, J., Corte-Real Santos, M., De Feo, L., Komada Eriksen, J., Hess, B., Kohel, D., Leroux, A., Longa, P., Meyer, M., Panny, L., Patranabis, S., Petit, C., Rodríguez-Henríquez, F., Schaeffler, S., Wesolowski, B.: SQISIGN: Algorithm specifications and supporting documentation. In: Proposal to NIST Standardization of Additional Digital Signature Schemes, Round1. National Institute for Standards and Technology (2023)
7. Chen, M., Imran, M., Ivanyos, G., Kutas, P., Leroux, A., Petit, C.: Hidden stabilizers, the isogeny to endomorphism ring problem and the cryptanalysis of pSIDH. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 99–130. Springer (2023)
8. Clavier, C., Feix, B., Gagnerot, G., Roussellet, M., Verneuil, V.: Square always exponentiation. In: Bernstein, D.J., Chatterjee, S. (eds.) Progress in Cryptology - INDOCRYPT 2011 - 12th International Conference on Cryptology in India, Chennai, India, December 11-14, 2011. Proceedings. Lecture Notes in Computer Science, vol. 7107, pp. 40–57. Springer (2011). https://doi.org/10.1007/978-3-642-25578-6_5, https://doi.org/10.1007/978-3-642-25578-6_5
9. Cornacchia, G.: Su di un metodo per la risoluzione in numeri interi dell'equazione $\sum_{h=0}^{n} C_h x^{n-h} y^h = P$. Giornale di Matematiche di Battaglini, 49:33-90 (1908)
10. Courrège, J., Feix, B., Roussellet, M.: Simple power analysis on exponentiation revisited. In: Gollmann, D., Lanet, J., Iguchi-Cartigny, J. (eds.) Smart Card Research and Advanced Application, 9th IFIP WG 8.8/11.2 International Conference, CARDIS 2010, Passau, Germany, April 14-16, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6035, pp. 65–79. Springer (2010). https://doi.org/10.1007/978-3-642-12510-2_6, https://doi.org/10.1007/978-3-642-12510-2_6
11. Couveignes, J.M.: Hard homogeneous spaces. IACR Cryptol. ePrint Arch. p. 291 (2006), http://eprint.iacr.org/2006/291
12. Dartois, P., Leroux, A., Robert, D., Wesolowski, B.: SQISignHD: new dimensions in cryptography. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 3–32. Springer (2024)
13. De Feo, L., Kohel, D., Leroux, A., Petit, C., Wesolowski, B.: SQISign: compact post-quantum signatures from quaternions and isogenies. In: ASIACRYPT 2020, Daejeon, South Korea, Part I 26. pp. 64–93. Springer (2020)
14. Deuring, M.: Die typen der multiplikatorenringe elliptischer funktionenkörper. Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg **14**, 197–272 (1941)
15. Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: CRYSTALS-Dilithium: A lattice-based digital signature scheme. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2018**(1), 238–268 (2018). https://doi.org/10.13154/tches.v2018.i1.238-268, https://doi.org/10.13154/tches.v2018.i1.238-268
16. Duparc, M., Fouotsa, T.B.: SQIPrime: A dimension 2 variant of SQISignHD with non-smooth challenge isogenies. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 396–429. Springer (2024)
17. Eriksen, J.K., Panny, L., Sotáková, J., Veroni, M.: Deuring for the people: Supersingular elliptic curves with prescribed endomorphism ring in general characteristic. Cryptology ePrint Archive (2023)
18. Feo, L.D., Leroux, A., Longa, P., Wesolowski, B.: New Algorithms for the Deuring Correspondence - Towards Practical and Secure SQISign Signatures. In: EUROCRYPT 2023. vol. 14008, pp. 659–690. Springer (2023). https://doi.org/10.1007/978-3-031-30589-4_23

19. Feo, L.D., Mrabet, N.E., Genêt, A., Kaluderovic, N., de Guertechin, N.L., Pontié, S., Tasso, É.: SIKE channels zero-value side-channel attacks on SIKE. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2022**(3), 264–289 (2022). https://doi.org/10.46586/TCHES.V2022.I3.264-289
20. Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Prest, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z.: FALCON: Fast-fourier lattice-based compactsignatures over ntru. In: Proposal to NIST Standardization of Additional Digital Signature Schemes. National Institute for Standards and Technology (2023)
21. Galbraith, S.D., Petit, C., Silva, J.: Identification protocols and signature schemes based on supersingular isogeny problems. In: ASIACRYPT 2017, Hong Kong, China, Part I 23. pp. 3–33. Springer (2017)
22. Hardy, G.H., Ramanujan, S.: The normal number of prime factors of a number n. Quaterly Journal of Mathematics, 48:76-92 (1917)
23. Homma, N., Miyamoto, A., Aoki, T., Satoh, A., Shamir, A.: Comparative power analysis of modular exponentiation algorithms. IEEE Trans. Computers **59**(6), 795–807 (2010). https://doi.org/10.1109/TC.2009.176, https://doi.org/10.1109/TC.2009.176
24. Joye, M., Olivier, F.: Side-channel analysis. In: van Tilborg, H.C.A., Jajodia, S. (eds.) Encyclopedia of Cryptography and Security, 2nd Ed, pp. 1198–1204. Springer (2011). https://doi.org/10.1007/978-1-4419-5906-5_516, https://doi.org/10.1007/978-1-4419-5906-5_516
25. Joye, M., Yen, S.: The montgomery powering ladder. In: Jr., B.S.K., Koç, Ç.K., Paar, C. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers. Lecture Notes in Computer Science, vol. 2523, pp. 291–302. Springer (2002). https://doi.org/10.1007/3-540-36400-5_22, https://doi.org/10.1007/3-540-36400-5_22
26. Kirschmer, M., Voight, J.: Algorithmic enumeration of ideal classes for quaternion orders. SIAM J. Comput. **39**, 1714–1747 (2010). https://doi.org/10.1137/080734467
27. Kocher, P.C.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: Koblitz, N. (ed.) Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings. Lecture Notes in Computer Science, vol. 1109, pp. 104–113. Springer (1996). https://doi.org/10.1007/3-540-68697-5_9, https://doi.org/10.1007/3-540-68697-5_9
28. 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)
29. Lee, J., Heo, D., Kim, H., Kim, G., Kim, S., Kim, H., Hong, S.: Fault attack on sqisign. In: Saarinen, M.O., Smith-Tone, D. (eds.) Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Oxford, UK, June 12-14, 2024, Proceedings, Part II. Lecture Notes in Computer Science, vol. 14772, pp. 54–76. Springer (2024). https://doi.org/10.1007/978-3-031-62746-0_3, https://doi.org/10.1007/978-3-031-62746-0_3
30. Longa, P.: sqisign-ec23 (2023). https://github.com/SQISign/sqisign-ec23 (2023)
31. Messerges, T.S., Dabbish, E.A., Sloan, R.H.: Power analysis attacks of modular exponentiation in smartcards. In: Koç, Ç.K., Paar, C. (eds.) Cryptographic Hardware and Embedded Systems, First International Workshop, CHES'99, Worcester, MA, USA, August 12-13, 1999, Proceedings. Lecture Notes in Computer Science,

vol. 1717, pp. 144–157. Springer (1999). https://doi.org/10.1007/3-540-48059-5_14, https://doi.org/10.1007/3-540-48059-5_14
32. Montgomery, P.L.: Speeding the pollard and elliptic curve methods of factorization. Mathematics of Computation **48**, 243–264 (1987), https://api.semanticscholar.org/CorpusID:4262792
33. Nakagawa, K., Onuki, H., Castryck, W., Chen, M., Invernizzi, R., Lorenzon, G., Vercauteren, F.: SQIsign2D-East: A new signature scheme using 2-dimensional isogenies. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 272–303. Springer (2024)
34. National Institute for Standards and Technology (NIST): Post-quantum cryptography: Additional digital signature schemes, https://csrc.nist.gov/projects/pqc-dig-sig/round-2-additional-signatures
35. National Institute for Standards and Technology (NIST): Call for proposals, post-quantum cryptography standardization (2016), https://csrc.nist.gov/csrc/media/projects/post-quantum-cryptography/documents/call-for-proposals-final-dec-2016.pdf (2016)
36. Park, D., Choi, M., Kim, G., Bae, D., Kim, H., Hong, S.: Stealing keys from hardware wallets: A single trace side-channel attack on elliptic curve scalar multiplication without profiling. IEEE Access **11**, 44578–44589 (2023). https://doi.org/10.1109/ACCESS.2023.3273150
37. Pope, G., Meyer, M., Santos, M.C., Eriksen, J.K.: Learningtosqi/sqisign-sagemath. https://github.com/LearningToSQI/SQISign-SageMath.git (2023)
38. Rostovtsev, A., Stolbunov, A.: Public-key cryptosystem based on isogenies. IACR Cryptology ePrint Archive **2006**, 145 (2006)
39. Shanks, D.: Five number-theoretic algorithms. Congressus Numerantium 7, Utilitas Mathematica pp. 51–70 (1973)
40. Silverman, J.H.: The arithmetic of elliptic curves, Graduate texts in mathematics, vol. 106. Springer (1986)
41. Tonelli, A.: Bemerkung uber die au osung quadratischer congruenzen. Gottinger Nachrichten pp. 344–346 (1891)
42. Voight, J.: Quaternion algebras. Springer Nature (2021)
43. Zhang, F., Yang, B., Dong, X., Guilley, S., Liu, Z., He, W., Zhang, F., Ren, K.: Side-Channel Analysis and Countermeasure Design on ARM-Based Quantum-Resistant SIKE. IEEE Trans. Computers **69**(11), 1681–1693 (2020). https://doi.org/10.1109/TC.2020.3020407