An attack on SIDH with arbitrary starting curve (draft)

Luciano Maino and Chloe Martindale

University of Bristol

August 25, 2022[*]

Abstract

We present an attack on SIDH which does not require any endomorphism information on the starting curve. Our attack has subexponential complexity thus significantly reducing the security of SIDH and SIKE; our analysis and preliminary implementation suggests that our algorithm will be feasible for the Microsoft challenge parameters $p = 2^{110}3^{67} - 1$ on a regular computer. Our attack applies to any isogeny-based cryptosystem that publishes the images of points under the secret isogeny, for example Séta [28] and B-SIDH [9]. It does not apply to CSIDH [8], CSI-FiSh [3], or SQISign [11].

1 Introduction

Supersingular Isogeny Diffie-Hellman (SIDH) [16] is a key exchange proposed in 2011 by Jao and De Feo that makes use of isogenies between elliptic curves. A well-studied hard problem in number theory is to find an unknown high-degree isogeny between two (supersingular) elliptic curves over a finite field, on which many cryptosystems [3, 8, 9, 11, 28] are based. This is a problem that is also believed to be hard for quantum computers, and as such *isogeny-based cryptography* has been one of the frontrunners in developing post-quantum cryptographic algorithms. Arguably the most influential primitive in the field of isogeny-based cryptography is Supersingular Isogeny Key Encapsulation (SIKE) [14], which is the incarnation of SIDH that was submitted to the NIST competition to find a new post-quantum-safe cryptographic standard [22] and is currently in the Fourth Round to be considered for standardization. In comparison to cryptosystems that rely purely on the isogeny problem, such as CSIDH [8], CSI-FiSh [3], and SQISign [11], the hardness assumption underlying SIKE is weaker as the image of some torsion points under the secret isogeny are also revealed. This gives rise to the *supersingular isogeny with torsion* (SSI-T) problem stated more precisely below. This has been shown to be weaker than the pure isogeny problem in a line of work pioneered by Petit [24] in 2017 and continued and built upon in multiple papers in the last 5 years [5, 13, 26]. However, the SIKE parameters had not been effected by these attacks, which all applied only to variants of SIDH.

In this paper we present an algorithm that solves the supersingular isogeny with torsion (SSI-T) problem in reasonable time for parameters that were believed to be secure, which is the hardness assumption underlying SIKE as well as any other SIDH-related protocols such as B-SIDH [9] and Séta [28]. This is recalled below:

[*]Supercedes [21] from 8th August 2022.

Supersingular Isogeny with Torsion (SSI-T):

Given coprime integers A and B, two supersingular elliptic curves $E_0/\mathbb{F}_{p^2}$ and $E_A/\mathbb{F}_{p^2}$ connected by an unknown degree A-isogeny $\varphi_A : E_0 \to E_A$, and given the restriction of φ to the B-torsion of E_0, recover an isogeny φ matching these constraints.

In particular, note that the SSI-T problem does not assume that E_0 is special in any way, for example that it has known endomorphism ring; our attack applies for any starting curve E_0. As such, it does not have the obvious mitigation that previous torsion-point attacks have had of using a trusted setup. Our attack has subexponential complexity and as such there will be large parameters for which our attacks become infeasible; however since releasing the first version of this paper [21] Damien Robert has released a polynomial-time attack on SSI-T that also does not assume that E_0 is special in any way [27].

Finally, our attack makes full use of the public torsion points and as such has no effect on any isogeny-based cryptosystem that does not publish images of points under the secret isogeny, such as CSIDH [8], CSI-FiSh [3], and SQISign [11].

Related work

The inspiration for this attack came from an unrelated collaboration of Luciano Maino with Wouter Castryck and Thomas Decru studying superspecial principally polarized abelian surfaces and (2,2)-isogenies between them, as well as endomorphisms of the form

$$\begin{pmatrix} a & b\widehat{\varphi} \\ c\varphi & d \end{pmatrix}.$$

Upon discovering our attack and sharing our idea with Castryck and Decru we found out that they had independently discovered an attack building on the previous collaboration in a different direction. They had already written their implementation and paper, although it was not yet public, and they were kind enough to share both with us as well as to note our forthcoming independent attack in their paper. In particular we were able to build on their implementation rather than starting from scratch when implementing the (2,2)-isogenies in our algorithm, for which we are very grateful. Their paper is now public [7]; they present a polynomial-time attack and provide an implementation for all the proposed NIST parameter sets for SIKE, but their attack relies on the knowledge of $\mathrm{End}(E_0)$, so can potentially be mitigated by generating a starting curve with no known non-scalar endomorphisms. Unsurprisingly given the common source of inspiration, our attack is similar to that of Castryck and Decru, especially the version they hint at in [7, §8.3].

One point where our attack differs is that we recover the secret directly rather than using a 'decision strategy' (c.f. [7, §2.3]); Rémy Oudompheng [23] and Benjamin Wesolowski [33] noted that the Castryck-Decru attack can be altered to recover the secret directly at a similar time to the first version of this paper becoming widely available.

Finally, since making this work available in [21], Damien Robert has significantly improved on the result [27], giving a polynomial-time attack on SIDH with arbitrary starting curve by constructing a special endomorphism of $E_0^4 \times E_A^4$.

Further acknowledgements

We would like to thank Lorenz Panny for useful discussions regarding complexity of extension field arithmetic and computations of isogenies with non-rational points in the kernel, for useful comments on an earlier draft of this paper, and for working on the implementation of the attack together with Giacomo Pope, to whom we are also deeply indebted. We thank Christophe Petit, for useful comments regarding methods to compute isogenies with irrational kernel points, and Eda Kirimli, for useful discussions. We are also extremely grateful to Luca De Feo, who shared with us a better method to find attack parameters during ANTS-XV, which in particular led to an argument that our algorithm has subexponential complexity. We would also like to thank COSIC and KU Leuven for hosting Luciano Maino as an intern, sparking his collaboration that led to this paper. Luciano Maino was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) Centre for Doctoral Training (CDT) in Trust, Identity, Privacy and Security in Large-scale Infrastructures (TIPS-at-Scale) at the Universities of Bristol and Bath.

Comparison to previous version

This is the second version of this paper; we note here the differences for the convenience of the reader. The most important difference is the statement of Theorem 1: In version one [21] we claimed that the (B, B)-isogeny with kernel

$$\langle (P_B, c\varphi(P_B)), (Q_B, c\varphi(Q_B)) \rangle$$

was an endomorphism of $E \times E'$. Upon working on the implementation of Algorithm 1 together with Lorenz Panny and Giacomo Pope (in particular implementing this (B, B)-isogeny) we observed that it is not an endomorphism but an isogeny as now stated correctly in Theorem 1. Algorithm 1 is basically unchanged except for pushing the A-torsion of E' through the isogeny Φ rather than the A-torsion of E.

There are additionally some minor changes: A new algorithm to select attack parameters from Luca De Feo answering [21, Open Question 2], an argument that this method of parameter selection leads to an attack of complexity $L_A(c, 1/2)$ for some constant c (also due to De Feo), and some updates on related work [23, 27, 33] representing advances that have appeared since [21].

2 The attack

Let all notation be as in the SSI-T problem statement above. The core idea behind our attack is to construct an elliptic curve E, an isogeny $\varphi_f : E \to E_0$, and a polarized isogeny Φ originating from the abelian surface $E \times E_A$ such that one of its components reveals the dual of the secret isogeny $\varphi_A \colon E_0 \to E_A$. This is a natural generalization of previous papers on torsion-point attacks (for example [24, 26]) in which a special endomorphism of E_A was constructed that leaked information about the secret isogeny; the success of using isogenies from $E \times E_A$ in place of elements of $\mathrm{End}(E_A)$ essentially stems from there being more choice in higher dimension.

We will notate our isogenies $E_1 \times E_2 \to E_3 \times E_4$ in matrix notation; the matrix

$$\begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix}$$

represents that $\alpha \in \mathrm{Hom}(E_1, E_3)$, $\beta \in \mathrm{Hom}(E_2, E_3)$, $\gamma \in \mathrm{Hom}(E_1, E_4)$, and $\delta \in \mathrm{Hom}(E_2, E_4)$, and maps

$$(P, P') \mapsto (\alpha(P) + \beta(P'), \gamma(P) + \delta(P')).$$

Our attack is a consequence of the following theorem:

Theorem 1. *Let f, A, and B be pairwise coprime integers such that $B = f + A$ and $c = 1/A \pmod{B}$, and let $E/\mathbb{F}_{p^2}$, $E'/\mathbb{F}_{p^2}$, $E_0/\mathbb{F}_{p^2}$, and $F/\mathbb{F}_{p^2}$ be elliptic curves connected by the following commutative diagram of isogenies:*

$$\begin{tikzcd}
E_0 \arrow[r, "\varphi'"] & E' \\
E \arrow[u, "\varphi_f"] \arrow[ur, "\varphi"] \arrow[r, "g'"'] & F, \arrow[u, "g_f"']
\end{tikzcd} \tag{1}$$

where $\deg(\varphi_f) = \deg(g_f) = f$ *and* $\deg(\varphi') = \deg(g') = A$*. Let* $\{P_B, Q_B\}$ *be a basis of* $E[B]$*, and let*

$$K \coloneqq \langle (P_B, c\varphi(P_B)), (Q_B, c\varphi(Q_B)) \rangle.$$

Then K is the kernel of the (B, B)-isogeny

$$\Phi = \begin{pmatrix} \varphi_f & -\widehat{\varphi}' \\ g' & \widehat{g}_f \end{pmatrix} \in \mathrm{Hom}(E \times E', E_0 \times F),$$

which respects the natural product polarizations on $E \times E'$ and $E_0 \times F$.

Before proving this theorem, we present our attack. For ease of notation, in all that follows we will assume that $A = \ell_A^a$ and $B = \ell_B^b$. We can use Theorem 1 with $(E', \varphi') = (E_A, \varphi_A)$ to recover φ_A by evaluating Φ on $E_A[A]$ and deducing the action of $-\widehat{\varphi}_A$ on $E_A[A]$; equivalently we can recover part of φ_A by choosing φ' that divides φ_A. However, to compute Φ we first need to find its domain; in particular, the attacker needs to compute φ_f. The attacker can choose any isogeny of degree f and an elliptic curve E (satisfying the conditions coming from Theorem 1) such that $\varphi_f : E \to E_0$. However, the problem faced by the attacker is that the computation of φ_f is not necessarily easy as there is no reason that $B - A$ would be smooth. To mitigate this, we increase our pool of available cofactors f by brute-forcing the last few steps of φ_A and/or by brute-forcing some extra torsion-point images.

The picture that we should keep in mind when reading through the attack below is the following commutative diagram, where:

- $\varphi_A : E_0 \to E_A$ is the secret key,
- $\varphi_f : E \to E_0$ is a f-isogeny chosen by the attacker,[1]
- $\varphi_{\ell_A^i} : E' \to E_A$ is a guess of the (dual of the) last i steps of φ_A,
- $\varphi' : E_0 \to E'$ is the corresponding first $a - i$ steps of φ_A such that $\varphi_A = \varphi_{\ell_A^i} \circ \varphi'$, and
- $\varphi : E \to E'$ is the $f\ell_A^{a-i}$-isogeny to which we apply Theorem 1.

$$\begin{tikzcd}
E_0 \arrow[r, "\varphi'"'] \arrow[rr, bend left, "\varphi_A"] & E' \arrow[r, "\varphi_{\ell_A^i}"'] & E_A \\
E \arrow[u, "\varphi_f"] \arrow[ur, "\varphi"'] & &
\end{tikzcd} \tag{2}$$

[1]In practise, the attacker computes $\widehat{\varphi}_f$ and deduces φ_f from this.

The attack is described in Algorithm 1. To discuss the complexity of this attack we should split it into three parts:

1. The precomputation step (Step 1); this can be done once-and-for-all for each parameter set A, B.
2. The cofactor isogeny computation (Step 2); if SIDH is set up with a fixed (arbitrary) E_0, this can be done once-for-all for this E_0.
3. The online steps (Steps 3 to 7); these steps need to be performed for every new public key.

The cost of the cofactor isogeny computation. The cofactor isogeny is deterministic and chosen by the attacker. As such, it does not need to be recomputed at any point because a wrong guess was made when brute-forcing. We compute the isogeny φ_f via a chain of prime-degree $q-$isogenies φ_q. The cost of computing φ_q for the larger factors of q is discussed in detail in Section 2.1.2 and can be approximated by $\tilde{O}(q^{3/2})$ multiplications in $\mathbb{F}_{p^2}$ plus a call to an oracle to find an irreducible polynomial of degree $\approx q$ in $\mathbb{F}_{p^2}$ or $\tilde{O}(q^2)$ multiplications in $\mathbb{F}_{p^2}$ plus a call to an oracle for factoring the $q-$division polynomial over $\mathbb{F}_{p^2}$.

The cost of the online steps. The discussion in Section 2.1.1 approximates the cost of Steps 3 to 7 by $\approx C \cdot e^4 \ell_A^i q_e^4 \log q_e$, where q_e is the largest prime dividing e and C is polynomial in $\log(p)$. We allow i and e to grow to increase the pool of options for f in order to get a smaller q_f.

The precomputation. If SIDH is set up to start every key exchange with a new E_0, the optimal choice of (e, i, j, f) for the attacker ensures that the cost of Step 2 is approximately the same as the cost of Steps 3 to 5. One could perform a brute force search over all parameters (e, i, j, f) such that $q_f^{3/2} \leq e^4 \ell_A^i q_e^4 \log q_e$ and $0 \leq j \leq b$, which would be costly.

However, since sharing the first version of this paper [21], Luca De Feo shared with us a subexponential algorithm for the precomputation leading both to a subexponential cofactor isogeny computation and to subexponential online steps. His argument is as follows: Suppose that we wish to target $A \approx B \approx 2^b$. To achieve subexponential complexity $L_{2^b}(c, 1/2)$, one can see from the complexity discussion of the online and cofactor steps above that it is sufficient to find parameters (e, i, j, f) such that $e, \ell_A^i \approx 2^{\sqrt{b}}$, and f is $\sqrt{b}^{\sqrt{b}}$-smooth.

To achieve this, we search for solutions to the equation

$$xA\ell_A^{-i} + yB\ell_B^{-j} = z, \tag{3}$$

where x and y are $\leq 2^{\sqrt{b}}$ and z is $\sqrt{b}^{\sqrt{b}}$-smooth, and i and j are fixed at some chosen values such that $\ell_A^i \approx \ell_B^j \approx 2^{\sqrt{b}}$. This corresponds to $e = -y$ (not necessarily coprime to B) and $f = -xz$; if xz, $y > 0$ then we switch the roles of A and B and this will correspond to $e = -x$ and $f = -yz$. Writing $f = -xz$ corresponds to decomposing $\varphi_f : E_f \to E_0$ into a degree-$(-z)$-isogeny $\varphi_{-z} : E_f \to E_0'$ and a degree-x-isogeny $\varphi_x : E_0' \to E_0$, and recovering $\varphi_A \circ \varphi_x$ by applying Algorithm 1 with $A = xA$, $E_0 = E_0'$, and $\varphi_A = \varphi_A \circ \varphi_x$.

To find such (x, y, z) for a given (i, j), we run Euclid's xgcd algorithm on $(A\ell_A^{-i}, B\ell_B^{-j})$ until we find (x_0, y_0, z_0) and (x_1, y_1, z_1) such that $x_i, y_i \approx 2^{\sqrt{b}/2}$; this should correspond to $z_i \approx 2^{b-\sqrt{b}/2}$. Then, search through all linear combinations $uz_0 + vz_1$ with $u, v \leq 2^{\sqrt{b}/2}$ and save the smoothest result; call this z. An integer (such as z) of size 2^b is $\sqrt{b}^{\sqrt{b}}$-smooth with probability $\rho(\beta)$, where $2^{b/\beta} = \sqrt{b}^{\sqrt{b}}$ and ρ is the Dickman-ρ function which can be approximated by $\rho(\beta) \approx \beta^{-\beta}$. Therefore, we are likely to find a $\sqrt{b}^{\sqrt{b}}$-smooth choice z if the number of choices for (u, v), that is $2^{\sqrt{b}}$, is $\approx \beta^\beta$.

A short calculation shows that

$$\log_2(\beta^\beta) = \sqrt{b}\left(1 + \frac{2 - 2\log_2\log_2 b}{\log_2 b}\right) \approx \log_2(2^{\sqrt{b}}).$$

We give some examples for concrete parameters in Section 2.1.1.

```
Algorithm 1: Recovering the secret isogeny.
Input: Coprime integers $A = \ell_A^a$ and $B = \ell_B^b$, two supersingular elliptic curves $E_0/\mathbb{F}_{p^2}$ and
       $E_A/\mathbb{F}_{p^2}$ connected by an unknown degree-$A$-isogeny $\varphi_A : E_0 \to E_A$, a basis
       $\{P_B, Q_B\}$ of $E_0[B]$, a basis $\{P_A, Q_A\}$ of $E_0[A]$, the image points $\varphi_A(P_B)$, $\varphi_A(Q_B)$.
Output: $\varphi_A : E_0 \to E_A$.
1 Compute integers $e$, $j$, $f$, and $i$ such that $e$ is small and smooth, $0 \le j \le b$, $f$ is smooth and
    positive, $i$ is small, $(A\ell_A^{-i})^{-1} = c \pmod{eB\ell_B^{-j}}$, and $eB\ell_B^{-j} = f + A\ell_A^{-i}$. For ease of
    notation, we set $A' = A\ell_A^{-i}$ and $B' = B\ell_B^{-j}$. For more details, see Section 2.1.1.
2 Compute a curve that is $f$-isogenous to $E_0$, define the dual of the computed isogeny to be
    $\varphi_f : E \to E_0$, and compute $\widehat{\varphi}_f(P_A), \widehat{\varphi}_f(Q_A), \widehat{\varphi}_f(P_B), \widehat{\varphi}_f(Q_B)$. For more details, see
    Section 2.1.2.
3 Compute a basis $\{P_{eB'}, Q_{eB'}\}$ of $E[eB']$ such that $[e]P_{eB'} = [\ell_B^j]\widehat{\varphi}_f(P_B)$ and
    $[e]Q_{eB'} = [\ell_B^j]\widehat{\varphi}_f(Q_B)$.
4 Choose a guess $\varphi_{\ell_A^i} : E' \to E_A$ for the last $i$ steps of $\varphi_A$, recall the definition of the
    corresponding $\varphi : E \to E'$ from diagram (4), and choose $R, S \in E'[eB']$ such that
        $$[e]R = [\ell_A^{-i} f \ell_B^j]\widehat{\varphi}_{\ell_A^i} \circ \varphi_A(P_B)$$
    and
        $$[e]S = [\ell_A^{-i} f \ell_B^j]\widehat{\varphi}_{\ell_A^i} \circ \varphi_A(Q_B);$$
    $R, S$ are a guess for the images $\varphi(P_{eB'}), \varphi(Q_{eB'})$ respectively.
5 Compute a $(eB', eB')$-isogeny with domain $E \times E'$ and kernel
        $$\ker(\Phi_{\text{guess}}) = \langle (P_{eB'}, cR), (Q_{eB'}, cS) \rangle.$$
    If the codomain splits, continue (see Remark 1). Else, return to Step 4 and take a new guess
    $(\varphi_{\ell_A^i}, R, S)$. For more details see Section 2.2.
6 Choose a basis $\{P, Q\}$ of $E'[A']$; compute $\widehat{\varphi'}(P)$ and $\widehat{\varphi'}(Q)$ via
        $$\Phi(0_E, P) = (-\widehat{\varphi'}(P), \widehat{g}_f(P))) \quad \text{and} \quad \Phi(0_E, Q) = (-\widehat{\varphi'}(Q), \widehat{g}_f(Q))).$$
7 Compute $\ker(\varphi') = \langle \widehat{\varphi'}(P), \widehat{\varphi'}(Q) \rangle$ and return $\varphi_{\ell_A^i} \circ \varphi'$.
```

Remark 1. *Step 5 in Algorithm 1 has a small chance of causing the overall algorithm to fail, as a split Jacobian may accidentally be the codomain for an incorrect guess. However it is easy to check whether or not E_0 is a factor and furthermore the chance of failure is very small.*

Proof of Theorem 1. To prove this result we employ Kani's Reducibility Criterion [17, Theorem 2.6] (c.f. similar use in [7, Theorem 1]). In Kani's language, we have set up our parameters so that $(\varphi, \ker(\varphi_f), [f^{-1}]\widehat{\varphi}_f \ker(\varphi'))$ is an *isogeny diamond configuration of order B*; that is, setting $H_1 = \ker(\varphi_f)$ and $H_2 = [f^{-1}]\widehat{\varphi}_f \ker(\varphi')$ we have the conditions $H_1 \cap H_2 = \{0_E\}$, $\#H_1 + \#H_2 = B$ and

$\#H_1 \cdot \#H_2 = \deg(\varphi)$. By [17, Theorem 2.6], there exists a unique anti-isometry $\psi\colon E[B] \to E'[B]$ such that, for all $P_i \in [B]^{-1}H_i$,

$$\psi(fP + AQ) = \varphi(Q - P).$$

Since in the rest of the proof we shall use ψ explicitly, we now reconstruct ψ. Define $\widetilde{\psi}\colon E[B] \to E'[B]$ to be the map $\widetilde{\psi}(P) \coloneqq c\varphi(P)$. The map $\widetilde{\psi}$ is an isomorphism and also an anti-isometry with respect to the B-Weil pairing:

$$e_B^{E'}(\widetilde{\psi}(P), \widetilde{\psi}(Q)) = e_B^E(P,Q)^{c^2fA} = e_B^E(P,Q)^{-c^2A^2} = e_B^E(P,Q)^{-1}.$$

Let $P_i \in [B]^{-1}H_i$, then $\phi(P_i) \in E'[B]$. Observe that

$$\widetilde{\psi}(fP + AQ) = c\varphi(fP + AQ) = \varphi(Q) - \varphi(P),$$

hence $\psi = \widetilde{\psi} = c\varphi_{|E[B]}$.

Furthermore, we have set up our parameters so that, following [17, Remark 2.2], the tuple $(\varphi, \varphi_f, \varphi', g', g_f)$ is an an *isogeny factor set representing* (φ, H_1, H_2). It then follows from the proof of [17, Theorem 2.6] that Φ is an isogeny $E \times E' \to E_0 \times F$ whose kernel is maximal isotropic with respect to the B-Weil pairing by identifying the principal polarization of each elliptic curve with the identity map.

Finally, by the proof of [17, Corollary 2.4], the kernel of Φ is given by the graph of $\psi = c\varphi_{|E[B]}$, that is $\ker(\Phi) = \langle (P_B, c\varphi(P_B)), (Q_B, c\varphi(Q_B)) \rangle$.

□

2.1 Complexity of Algorithm 1

Here we give some details on and study the complexity of the first four steps of Algorithm 1 in the case relevant to SIKE, namely $A = 3^a$ and $B = 2^b$, with a focus on the Microsoft challenge parameters $A = 3^{67}$ and $B = 2^{110}$ and the parameters that were proposed for NIST level I $A = 3^{137}$ and $B = 2^{216}$.

2.1.1 Choosing parameters

To understand Step 1, we recall the commutative diagram that we keep in mind during this attack, where:

- $\varphi_A : E_0 \to E_A$ is the secret key,
- $\varphi_f : E \to E_0$ is a f-isogeny chosen by the attacker,
- $\varphi_{\ell_A^i} : E' \to E_A$ is a guess of the last i steps of φ_A,
- $\varphi' : E_0 \to E'$ is the corresponding first $a - i$ steps of φ_A such that $\varphi_A = \varphi_{\ell_A^i} \circ \varphi'$, and
- $\varphi : E \to E'$ is the $f\ell_A^{a-i}$-isogeny to which we apply Theorem 1.

$$\begin{tikzcd}
E_0 \arrow[r, "\varphi'"'] \arrow[rr, bend left, "\varphi_A"] & E' \arrow[r, "\varphi_{\ell_A^i}"'] & E_A \\
E \arrow[u, "\varphi_f"] \arrow[ur, "\varphi"'] & &
\end{tikzcd} \tag{4}$$

Choosing f. The shape of f determines the complexity of computing φ_f. The cofactor f does not need to be small as the isogeny is deterministic but it does need to be smooth: consider the extreme case that f is prime and $\approx A$, computing φ_f directly will be harder than computing φ_A directly (because of the extension field arithmetic). Exactly how smooth we require f to be depends on what we hope we can achieve in complexity for the attack. If q is the largest prime divisor of f then the complexity of Step 2 will be dominated by the cost of the computation of a q-isogeny, which can be performed $O(\sqrt{q})$ multiplications in the field of definition of a generator of the kernel of the isogeny using sqrtVelu [1, Section 4.1.4]. The field of definition is however hard to control, and large field extensions can seriously slow down our arithmetic. It is hard to make this precise; for some values of q the minimal k for which $E(\mathbb{F}_{p^k})$ contains a q-torsion point will be much smaller than q and in some cases it will be much larger.

In order to make an approximation of the complexity of computing φ_f on which we can base our search for good parameters for our attack, we ran some experiments to look at the behaviour of field extensions for different values of p. As an illustration let us consider $E_{1728}/\mathbb{F}_p$ with $p = 2^{216}3^{137} - 1$ as in the proposed NIST level I parameters for SIKE. Only the even degree extension fields are relevant as the codomain of each isogeny is defined over $\mathbb{F}_{p^2}$. Figures 2, 3, and 4 show the q for which there exists an even $k \leq 1000$ such that there is an $\mathbb{F}_{p^k}$-rational point of order q (only the minimal even k is depicted). In total, we find 72% of the primes $< 10^2$ (c.f. Figure 2), 62.5% of the primes $< 10^3$ (c.f. Figure 3), and 22% of the primes $< 10^4$ (c.f. Figure 4). Based on these experiments, to guide our parameter selection for our attack we make a very crude estimate that we expect the minimal field extension for the maximal q dividing f is about size q. This gives us a very rough estimate of $\tilde{O}(q^{3/2})$ for the complexity of computing a φ_q-isogeny, which in turn is the dominating cost of computing φ_f. If the largest factor of our smoothest f only admits very large extension fields we can of course choose to take a slightly less smooth f (that is, a slightly bigger q) where the field extension is smaller, or use Kohel's algorithm at the expense of increase the complexity to $\tilde{O}(q^2)$; see Section 2.1.2 for more details.

Choosing i and e. The cost coming from i is the cost of brute-forcing all the cyclic $3^i = \ell_A^i$-isogenies from E_A, which costs $\approx 3^i$ multiplications in $\mathbb{F}_{p^2}$. This is however multiplied by the brute-force cost of guessing the images of the e-torsion points in Step 4 and by the cost of computing Φ. Guessing the images of the e-torsion points amounts to checking all the pairs of points of order e on E', which is $\approx e^4$. This is one sense in which e and i have to be 'small': We have to run Steps 3 to 5 of Algorithm 1 $\approx e^4 3^i$ times.

Additionally, the isogeny Φ (which we will attempt to compute $\approx e^4 3^i$ times) is an (eB', eB')-isogeny; in particular it factors via an (e, e)-isogeny. So, in addition we require e to be q-smooth, where q is the largest prime for which it is feasible to compute (q, q)-isogenies (potentially over an extension field, which again will add a non-negligible cost). The need for the computation of the (e, e)-isogeny is the main barrier to implementing our algorithm for the proposed NIST parameters, as to do so requires a working implementation of (q, q)-isogenies, which while should theoretically be possible and reasonably fast, requires some research to achieve. There exists literature on this topic [4, 6, 19, 20], from which we have made a baseline assumption than computation of a (q, q)-isogeny over $\mathbb{F}_{p^k}$ can be performed in $O(q^3)$ multiplications in $\mathbb{F}_{p^k}$. However, there is very little existing work in the way of practical implementation of supersingular Jacobians and products of elliptic curves. We do note here that it would be possible to avoid implementing the factors of the (e, e)-isogenies to also map to and from products of elliptic curves, as we can ensure to start and finish the computation of Φ with a (2,2)-isogeny, which may make the practical implementation of (e, e)-isogenies with regards to this attack a more achievable goal.

Working with our baseline assumption that a (q, q)-isogeny can be computed in approximately q^3 multiplications over the base field of its kernel, we expect the cost of computing Φ as a (eB, eB)-isogeny to be dominated by the cost of computing a (q, q)-isogeny where q is the largest prime factor of e. We leave a careful analysis of the sizes of the field extensions for genus 2 to later work that

Figure 1: Extension field degrees < 1000 needed for $\mathbb{F}_{p^k}$-rational q-torsion

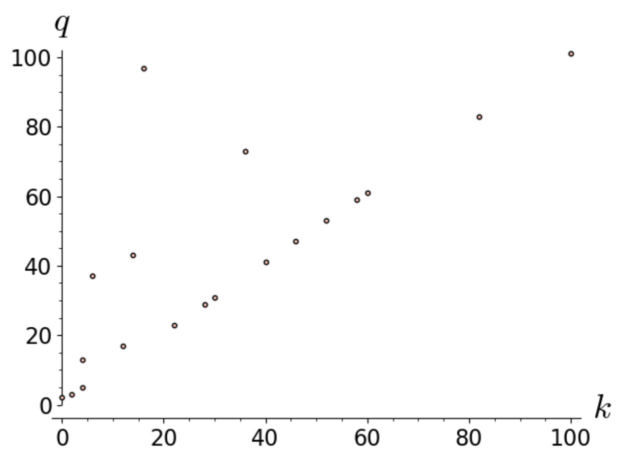

Figure 2: $q < 10^2$

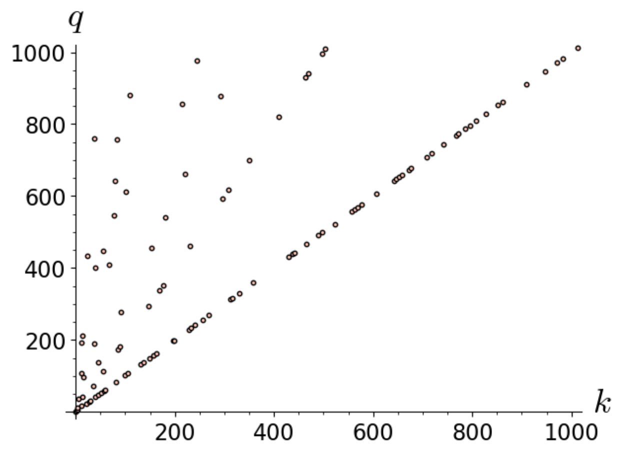

Figure 3: $q < 10^3$

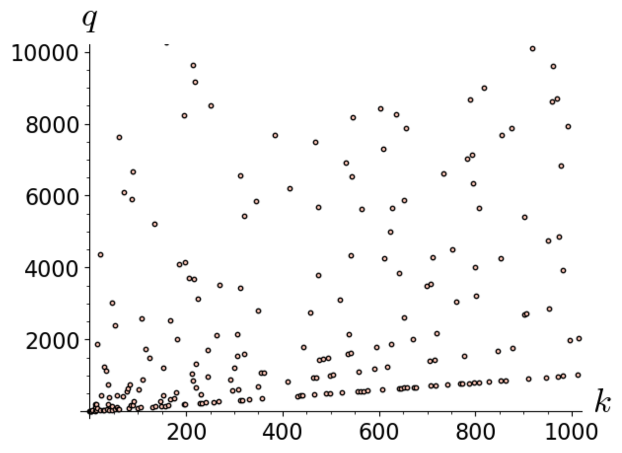

Figure 4: $q < 10^4$

includes a practical implementation of (q, q)-isogenies for prime $q \neq 2$, but let us assume for the sake of argument that the slow down for the extension field arithmetic scales with q similarly to the elliptic curve case. Then we approximate the cost of computing the (q, q)-isogeny by $O(q^3 \cdot q \log q)$. This is probably an overestimate: More research is needed into the existence of sqrtVélu-style-algorithms in the case of abelian surfaces. However, if the attack costs 2^λ, note that e is already forced to be relatively small compared to this by the fact that we have to search through $\approx e^4$ pairs of possible images of e-torsion points. Because of this, we can expect e to be fairly smooth compared to f, for example, so q (and the corresponding field extension) need not be particularly large.

In our choice of parameters for our toy example, we have chosen to demonstrate the use of e without the need to delve into (q, q)-isogenies for $q > 2$ by choosing $e = 2^3$. In this case we need a field extension of degree 4 for the 2^{b+1}-torsion points, of degree 8 for the 2^{b+2}-torsion points and of degree 16 for the 2^{b+3}-torsion points. This is not special to this instance but a consequence of the fact that the pull-back of the multiplication-by-2 map contains a square root (and no other rational but not integral powers), and so each lift of a point of order 2^i to a point of order 2^{i+1} will either double the degree of the field extension or keep it the same.

Choosing j. The choice of j only potentially effects the precomputation step, Step 1 of Algorithm 1, as we achieve $B' = 2^{-j}B$-torsion points by multiplying the known B-torsion by 2^j; for this reason we have no restrictions on non-negative j. Notice that we do not require e to be coprime to B, so e may contain powers of two, accounting also for the possibility of negative j.

Concrete attack parameters. We present here some choices of attack parameters in three cases of interest: A toy example to test our algorithm, the Microsoft challenge parameters, and the parameters of SIKEp434 that were proposed for NIST Level I.

Toy parameters: We consider a small example to test our algorithm: $A = 3^8$, $B = 2^{15}$, $i = 3$, $e = 8$, $j = 0$, $f = 5 \cdot 7^2 \cdot 11 \cdot 97$. The largest field extension that we need for the computation of φ_f is $\mathbb{F}_{p^{20}}$, for the 11-isogeny. The largest field extension for $e = 8$ is 16, for the pullbacks of the order-2^{15} points to order-2^{18} points.

Challenge parameters: We consider one of the sets of challenge parameters put forward by Microsoft [10]: $A = 3^{67}$, $B = 2^{110}$, $i = 7$, $e = 1$, $j = 2$,

$$f = 5 \cdot 7 \cdot 13^3 \cdot 43^2 \cdot 73 \cdot 151 \cdot 241 \cdot 269 \cdot 577 \cdot 613 \cdot 28111 \cdot 321193.$$

The largest field extension we would need for the computation of φ_{321193} using sqrtVélu is of degree 642384; in this case it might be faster to use a variant of Kohel's algorithm to avoid the extension field arithmetic (see Section 2.1.2 for more details). Based on preliminary SageMath experiments, we expect the computation of φ_{321193} to be feasible on a regular laptop. The extension field degrees for all the factors of f are given by

$$\begin{aligned}[k, q] =& [8, 5], [12, 7], [24, 13], [28, 43], [144, 73], [75, 151], [480, 241], \\ & [67, 269], [1152, 577], [1224, 613], [56220, 28111], [642384, 321193].\end{aligned}$$

The choice of $i = 7$ also means that we need to run Steps 3 to 5 of Algorithm 1 up to $3^7 \approx 2^{11}$ times, which we expect to take at most a few hours on a laptop. In particular, if the SIDH instantiation uses a fixed (arbitrary) starting curve, the computation of φ_f can be performed as a precomputation and the attack on an individual public key is relatively fast, just some Richelot isogenies of abelian surfaces and 3-isogenies of elliptic curves, repeated potentially 3^7 times.

We have thus far restricted ourselves to e and B being a powers of two, as we want to demonstrate our attack and do not yet have adequate resources at our disposal to compute (ℓ, ℓ)-isogenies for

i	j	x	y	z
19	27	$41 \cdot 2333$	$-101 \cdot 241$	$-5^4 \cdot 19 \cdot 47 \cdot 61 \cdot 857 \cdot 2903 \cdot 60889 \cdot 216617$ $\cdot 342497 \cdot 2309969 \cdot 2945407 \cdot 3951767 \cdot 4037069$
16	24	1823581	$-239 \cdot 6553$	$-11 \cdot 13 \cdot 19 \cdot 29 \cdot 631 \cdot 6043 \cdot 16451 \cdot 29759 \cdot 139987$ $\cdot 364513 \cdot 1850837 \cdot 3464849 \cdot 6344729 \cdot 26440207$
15	27	123551	-2546657	$-5^2 \cdot 29 \cdot 103 \cdot 1549 \cdot 28201 \cdot 55933 \cdot 243431$ $\cdot 1874903 \cdot 4421117 \cdot 6553021 \cdot 14183149 \cdot 39691591$
16	29	$5 \cdot 7^2 \cdot 1171$	-7884713	$-173 \cdot 853 \cdot 883 \cdot 8627 \cdot 26759 \cdot 692929 \cdot 3500557$ $\cdot 5202137 \cdot 6065333 \cdot 15108221 \cdot 28512793$
16	25	$79 \cdot 139 \cdot 499$	$-197 \cdot 47777$	$-5 \cdot 11 \cdot 17 \cdot 571 \cdot 35099 \cdot 40639 \cdot 48889 \cdot 81281$ $\cdot 138899 \cdot 1285429 \cdot 8464307 \cdot 13664309 \cdot 17314859$
16	24	$-467 \cdot 5419$	$5 \cdot 434689$	$-7 \cdot 103 \cdot 109 \cdot 2791 \cdot 3643 \cdot 36191 \cdot 47581 \cdot 99817$ $\cdot 401119 \cdot 749467 \cdot 2690497 \cdot 2863607 \cdot 3014203$
16	25	$-197 \cdot 9391$	$11 \cdot 307 \cdot 941$	$-5 \cdot 233 \cdot 431 \cdot 659 \cdot 4219 \cdot 237277 \cdot 371341 \cdot 820643$ $\cdot 2362589 \cdot 3896323 \cdot 14204429 \cdot 55510211$
17	26	-1	1	$-11 \cdot 23 \cdot 31 \cdot 131 \cdot 281 \cdot 311 \cdot 601 \cdot 3331 \cdot 8059$ $\cdot 8761 \cdot 163411 \cdot 1164091 \cdot 2101681 \cdot 4027511 \cdot 11144321$

Table 1: Some possible attack parameters for SIKEp434

$\ell > 2$. However, looking at the Microsoft challenge parameters can already illustrate the freedom that being able to compute efficiently (ℓ, ℓ)-isogenies for $\ell \neq 2$ can provide: We open up more options for attack parameters, including in this case in which one requires very little brute-force (only repeating Steps 4 to Step 5 up to 4 times): $A = 2^{110}$, $B = 3^{67}$, $A' = 2^{a-j} = 2^{108}$, $B' = 3^{b-i} = 3^{48}$, $e = 1$, and

$$f = 5 \cdot 7 \cdot 13 \cdot 61 \cdot 73 \cdot 431 \cdot 593 \cdot 607 \cdot 881 \cdot 36997 \cdot 139393 \cdot 227233.$$

The extension field degrees for all the factors of f are given by

$$\begin{aligned}[k, q] =& [8, 5], [12, 7], [24, 13], [60, 61], [144, 73], \\ & [860, 431], [1184, 593], [303, 607], [220, 881], \\ & [73992, 36997], [34848, 139393], [56808, 227233].\end{aligned}$$

NIST Level I parameters: To select attack parameters for SIKEp434, that is, with $A = 3^{137}$ and $B = 2^{216}$, we rely on De Feo's algorithm for parameter selection outlined in the 'precomputation step' complexity analysis of Section 2. Table 1 shows some outputs of De Feo's algorithm for SIKEp434 parameters; these represent (i, j, x, y, z) such that

$$x3^{137-i} + y2^{216-j} = z.$$

We leave the details on the best parameter choice to further study, as all these parameters require a working implementation of (ℓ, ℓ)-isogenies for $\ell > 2$. Note that the last entry in the table only requires the computation of $(3, 3)$-isogenies, at the expense of some smoothness of $f = -yz$; the largest degree of elliptic-curve isogeny required in this choice is 11144321.

2.1.2 Computing the cofactor isogeny

The points in the kernel of a factor φ_q of φ_f will not be defined over $\mathbb{F}_{p^2}$ in general. When we choose the value of f, as well as checking that f is smooth, we check, for each prime factor q, the degree k of the field extension that would be required to find a point of order q.

When computing an isogeny $\varphi_q : E_n \to E_{n+1}$ through which φ_f factors, in order to control field extensions in the whole attack, we need to choose φ_q so that

- the codomain E_{n+1} is defined over $\mathbb{F}_{p^2}$, and
- the image points $\varphi_q(P)$ and $\varphi_q(Q)$ are defined over $\mathbb{F}_{p^2}$.

To compute large-degree isogenies, we can use the sqrtVélu method described in [1, Section 4.14], which has complexity $\tilde{O}(q^{1/2}\mathsf{m}_k)$, where m_k is the cost of a multiplication in $\mathbb{F}_{p^k}$. As outlined above we expect $k \approx q$ on average if we allow for some freedom in the choice of f. To guide our choice of attack parameters, we therefore take the complexity of computing our large-degree isogenies and the images of points under these to be $\tilde{O}(q^{3/2})$. However, for large k, finding an irreducible polynomial to generate $\mathbb{F}_{p^k}$ may be a bottleneck. We leave investigation into whether or not this search can be improved using a quantum algorithm to future work.

When the minimal extension degree of the field in which the kernel points of a q-isogeny φ_q are defined is large, it will be faster to instead use a variant of Kohel's algorithm [18, Section 2.4]. Kohel's algorithm computes the isogeny from its kernel polynomial, which, assuming E is defined as $y^2 = f(x)$, is defined by

$$K(x) = \prod_{(x_P, \pm y_P) \in \ker(\varphi)} (x - x_P) \in \mathbb{F}_{p^2}[x];$$

each x_P appears only once so $\deg(K) = (q-1)/2$. Constructing the kernel polynomial from this definition would also require computing the extension field in which the x_P live, but we can also construct a choice for $K(x)$ from the q-division polynomial.

The q-division polynomial for E is defined by

$$\psi_q(x) = \prod_{(x_P, \pm y_P) \in E[q]} (x - x_P) \in \mathbb{F}_{p^2}[x],$$

and can either be precomputed for an E with general coefficients (e.g. a Montgomery coefficient A) or computed recursively for a given E [29, Exercise 3.7]. In [2, Section 9] a careful analysis is given of both approaches to computing division polynomials; evaluation of a precomputed polynomial can be faster if q is fairly small but if q is large enough that multiplying polynomials of degree q^2 will be faster using FFT, then it will be faster to compute them directly for any given E. For these large q, the cost of computing the division polynomial is $O(q^2 \log q)$.

So, suppose we have computed the q-division polynomial, we can then factorize the degree-$(q^2-1)/2$-polynomial ψ_q into irreducible factors over $\mathbb{F}_{p^2}$. If there exists an irreducible factor of φ_q of degree $(q-1)/2$, then we can choose this for $K(x)$, the kernel polynomial of E, and compute φ_q using Kohel's algorithm in time $O(q^2)$. The factorization of division polynomials has been completely described by Verdure [32]. In particular, for large k there exists at least one irreducible factor of degree $(q-1)/2$ over $\mathbb{F}_{p^2}$.

Although factorization of large-degree polynomials is polynomial-time in $\log(p)$ and q, as q is large the complexity of this step can easily grow to infeasible. We hope that our algorithm can be improved by either searching for only one[2] irreducible factor of degree $(q-1)/2$, or by using quantum algorithms (e.g. [12]), or both. We leave the details of this to future work.

[2]Since we only need an irreducible factor of degree $(q-1)/2$, after the "Distinct-degree factorization" stage in [12], we can run "Equal-degree factorization" on a single square-free polynomial. However, this does not improve the asymptotic complexity of the algorithm in our case.

In this case, for large q, the entire cost of computing φ_q is $\tilde{O}(q^2)$ multiplications in $\mathbb{F}_{p^2}$, plus a call to a (quantum?) oracle for factoring.

2.2 Computing (ℓ, ℓ)-isogenies

In order for our algorithm to reach its full potential it is necessary to also consider integers e in Step 1 of Algorithm 1 that do not divide B, and in particular are not necessarily powers of two. It may also be that there is a nice parameter choice (e, i, j, f) with A a power of 2 and B a power of 3 (c.f. the attack parameter suggestions in Section 2.1.1), or one may want to consider more general setups. In all of these cases, in Step 5 of Algorithm 1 it will be necessary to compute (ℓ, ℓ)-isogenies for $\ell \neq 2$, which as observed above requires more research to achieve practically (for $\ell = 3$ there is however already some interesting work on this topic [6]). For this reason, we leave all instantiations of the attack that use e not dividing B to future work and focus on the case of $(2, 2)$-isogenies, that is, $B = 2^b$ and $e|B$. Recall that we set $B' = B2^{-j}$, where $0 \leq j \leq b$.

In order to compute the chain of $(2, 2)$-isogenies whose composition is the (eB', eB')-isogeny Φ, we need to able to compute three different flavours of $(2, 2)$-isogenies between principally polarized abelian surfaces:

- A (2,2)-isogeny from a Jacobian of a genus 2 curve to a Jacobian of a genus 2 curve, for which we refer to reader to [31, §2.3.1].
- A (2,2)-isogeny from a Jacobian of a genus 2 curve to a product of elliptic curves, for which we refer the reader to [30, Proposition 8.3.1]. (This is required for the last step of Φ).
- A (2,2)-isogeny from a product of elliptic curves to the Jacobian of a genus 2 curve, for which we refer the reader to [7] for more details. (This is required for the first step of Φ).

Upon sharing our application to breaking SIKE with Castryck and Decru, they were kind enough to share an implementation of this step with us that they had written for their paper [7], which at the time was not publicly available. Their implementation and description of the first step of the (2,2)-isogeny path (ProdToJac) and of the intermediate steps (JacToJac) provided us with useful insights, and the current (still unavailable) incarnation of our implementation of our attack now uses Giacomo Pope and collaborator's SageMath implementation [25] for these steps, modified only to include the computation of the images of the 3-power torsion points.

3 Future work

Our algorithm to attack SIDH, discovered independently from the polynomial-time attack of Castryck and Decru [7] but inspired by an earlier joint project, makes no use of any special endomorphisms on E_0 and as such can be applied to an arbitrary starting curve E_0. Our attack has complexity $L_A(c, 1/2)$, where c is some constant, however, since publishing the first version of this paper [21], Damien Robert has published an algorithm loosely inspired by our paper and [7] which breaks arbitrary-starting-curve SIKE in polynomial-time.

An implementation of our attack for the Microsoft challenge parameters will appear in a later version of this paper, and if progress is made on the computation of (ℓ, ℓ)-isogenies also for the proposed NIST level I parameters, to get some benchmark timings.

Finally, we propose some open questions, the answers to which will increase our understanding of the reach of this attack and how to compute rescaled parameters for SIKE achieving the required security levels with respect to this attack, assuming of course that it is possible to construct starting curves with unknown endomorphisms to mitigate the polynomial-time attack of [7].

Open question 1. How can we compute the optimal choice for f, taking into account all speed-ups available from subfield arithmetic? Can we implement an operation counter to find best trade-off?

Open question 2. The algorithm we currently use to select parameters in Step 1 of Algorithm 1, suggested by Luca De Feo, has subexponential complexity. Can we improve this algorithm?

Open question 3. For a given new large parameter set for SIKE, starting from a curve with no known non-integral endomorphisms, can we prove that it has 2^λ-bit security? That is, can we prove that there is no choice (e, i, j, f) for which our attack takes $< 2^\lambda$ multiplications in $\mathbb{F}_{p^2}$ (or equivalent)?

References

[1] D. J. Bernstein, L. D. Feo, A. Leroux, and B. Smith, "Faster computation of isogenies of large prime degree," https://eprint.iacr.org/2020/341, 2020. [Online]. Available: https://eprint.iacr.org/2020/341.

[2] D. J. Bernstein, T. Lange, C. Martindale, and L. Panny, "Quantum circuits for the csidh: Optimizing quantum evaluation of isogenies," in *Advances in Cryptology – EUROCRYPT 2019*, Y. Ishai and V. Rijmen, Eds., Cham: Springer International Publishing, 2019, pp. 409–441.

[3] W. Beullens, T. Kleinjung, and F. Vercauteren, "CSI-FiSh: Efficient isogeny based signatures through class group computations," in *Advances in Cryptology – ASIACRYPT 2019*, S. D. Galbraith and S. Moriai, Eds., Springer, 2019, pp. 227–247, ISBN: 978-3-030-34578-5.

[4] G. Bisson, R. Cosset, and D. Robert, *AVIsogenies MAGMA package*, https://gitlab.inria.fr/roberdam/avisogenies.

[5] P. Bottinelli, V. de Quehen, C. Leonardi, A. Mosunov, F. Pawlega, and M. Sheth, *The dark SIDH of isogenies*, IACR Cryptology ePrint Archive 2019/1333, https://ia.cr/2019/1333, 2019.

[6] R. Bröker, E. W. Howe, K. E. Lauter, and P. Stevenhagen, "Genus-2 curves and jacobians with a given number of points," *LMS Journal of Computation and Mathematics*, vol. 18, no. 1, pp. 170–197, 2015. DOI: 10.1112/S1461157014000461.

[7] W. Castryck and T. Decru, *An efficient key recovery attack on sidh (preliminary version)*, Cryptology ePrint Archive, Paper 2022/975, https://eprint.iacr.org/2022/975, 2022. [Online]. Available: https://eprint.iacr.org/2022/975.

[8] W. Castryck, T. Lange, C. Martindale, L. Panny, and J. Renes, "CSIDH: An efficient post-quantum commutative group action," in *ASIACRYPT (3)*, ser. Lecture Notes in Computer Science, https://ia.cr/2018/383, vol. 11274, Springer, 2018, pp. 395–427.

[9] C. Costello, "B-SIDH: Supersingular Isogeny Diffie–Hellman using twisted torsion," in *ASIACRYPT (2)*, ser. Lecture Notes in Computer Science, https://ia.cr/2019/1145, vol. 12492, Springer, 2020, pp. 440–463.

[10] ——, "The case for SIKE: A decade of the supersingular isogeny problem.," in *The NIST 3rd Post-Quantum Cryptography Standardization Conference.*, https://ia.cr/2021/543, 2021.

[11] L. De Feo, D. Kohel, A. Leroux, C. Petit, and B. Wesolowski, "SQISign: Compact post-quantum signatures from quaternions and isogenies," in *Advances in Cryptology – ASIACRYPT 2020*, S. Moriai and H. Wang, Eds., Cham: Springer International Publishing, 2020, pp. 64–93.

[12] J. Doliskani, "Toward an optimal quantum algorithm for polynomial factorization over finite fields," *Quantum Info. Comput.*, vol. 19, no. 1–2, pp. 1–13, Feb. 2019, ISSN: 1533-7146.

[13] T. B. Fouotsa, P. Kutas, S. Merz, and Y. B. Ti, "On the isogeny problem with torsion point information," Lecture Notes in Computer Science, vol. 13177, pp. 142–161, 2022, https://ia.cr/2021/153.

[14] D. Jao, R. Azarderakhsh, M. Campagna, C. Costello, L. De Feo, B. Hess, A. Hutchinson, A. Jalali, K. Karabina, B. Koziel, B. LaMacchia, P. Longa, M. Naehrig, G. Pereira, J. Renes, V. Soukharev, and D. Urbanik, "Supersingular isogeny key encapsulation," *Updated version of [15] for round 4 of [22]*, 2020.

[15] D. Jao, R. Azarderakhsh, M. Campagna, C. Costello, L. De Feo, B. Hess, A. Jalali, B. Koziel, B. LaMacchia, P. Longa, M. Naehrig, J. Renes, V. Soukharev, and D. Urbanik, "Supersingular isogeny key encapsulation," *Submission to [22]*, 2017, https://sike.org.

[16] D. Jao and L. De Feo, "Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies," in *PQCrypto*, ser. Lecture Notes in Computer Science, https://ia.cr/2011/506, vol. 7071, Springer, 2011, pp. 19–34.

[17] E. Kani, "The number of curves of genus two with elliptic differentials.," *Journal für die reine und angewandte Mathematik (Crelles Journal)*, vol. 1997, pp. 122–93, 1997.

[18] D. Kohel, "Endomorphism rings of elliptic curves over finite fields," http://iml.univ-mrs.fr/~kohel/pub/thesis.pdf, Ph.D. dissertation, University of California at Berkeley, 1996.

[19] D. Lubicz and D. Robert, "Fast change of level and applications to isogenies," in *Algorithmic Number Theory Symposium – ANTS-XV*, https://people.maths.bris.ac.uk/~jb12407/ANTS-XV/papers/ANTS-XV_lubicz-robert.pdf, 2022.

[20] D. Lubicz and A. Somoza, *AVIsogenies SageMath package*, https://gitlab.inria.fr/roberdam/avisogenies/-/tree/sage.

[21] L. Maino and C. Martindale, *An attack on sidh with arbitrary starting curve*, Cryptology ePrint Archive, Paper 2022/1026, Version 1: https://eprint.iacr.org/archive/2022/1026/20220808:211318, 2022.

[22] National Institute of Standards and Technology, *Post-quantum cryptography standardization*, https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Post-Quantum-Cryptography-Standardization, Dec. 2016.

[23] R. Oudompheng, *A note on implementing direct isogeny determination in the Castryck-Decru SIKE attack*, https://www.normalesup.org/~oudomphe/textes/202208-castryck-decru-shortcut.pdf.

[24] C. Petit, "Faster algorithms for isogeny problems using torsion point images," in *ASIACRYPT (2)*, ser. Lecture Notes in Computer Science, https://ia.cr/2017/571, vol. 10625, Springer, 2017, pp. 330–353.

[25] G. Pope, *Et. al., castryck-Decru key recovery attack on SIDH (SageMath implementation)*, https://github.com/jack4818/Castryck-Decru-SageMath, 2022.

[26] V. de Quehen, P. Kutas, C. Leonardi, C. Martindale, L. Panny, C. Petit, and K. E. Stange, "Improved torsion-point attacks on SIDH variants," in *Advances in Cryptology – CRYPTO 2021*, https://ia.cr/2020/633, 2021, pp. 432–470.

[27] D. Robert, *Breaking sidh in polynomial time*, Cryptology ePrint Archive, Paper 2022/1038, https://eprint.iacr.org/2022/1038, 2022.

[28] C. D. de Saint Guilhem, P. Kutas, C. Petit, and J. Silva, *SÉTA: Supersingular encryption from torsion attacks*, IACR Cryptology ePrint Archive 2019/1291, https://ia.cr/2019/1291, 2019.

[29] J. H. Silverman, *The arithmetic of elliptic curves.* Springer Science & Business Media, 2009, vol. 106.

[30] B. Smith, "Explicit endomorphisms and correspondences," Ph.D. dissertation, University of Sydney, 2005.

[31] Y. B. Ti, "Isogenies of Abelian Varieties in Cryptography," Ph.D. dissertation, University of Auckland, 2019.

[32] H. Verdure, "Factorisation patterns of division polynomials," *Proceedings of the Japan Academy, Series A, Mathematical Sciences*, vol. 80, no. 5, pp. 79–82, 2004. DOI: 10.3792/pjaa.80.79. [Online]. Available: https://doi.org/10.3792/pjaa.80.79.

[33] B. Wesolowski, *Understanding and improving the Castryck-Decru attack on SIDH*, https://mycore.core-cloud.net/index.php/s/iW1psgMzoo8gFqe#pdfviewer.