B-SIDH: SUPERSINGULAR ISOGENY DIFFIE-HELLMAN USING TWISTED TORSION

CRAIG COSTELLO

Abstract. This paper explores a new way of instantiating isogeny-based cryptography in which parties can work in both the $(p+1)$-torsion of a set of supersingular curves and in the $(p-1)$-torsion corresponding to the set of their quadratic twists. Although the isomorphism between a given supersingular curve and its quadratic twist is not defined over $\mathbb{F}_{p^2}$ in general, restricting operations to the x-lines of both sets of twists allows all arithmetic to be carried out over $\mathbb{F}_{p^2}$ as usual. Furthermore, since supersingular twists always have the same $\mathbb{F}_{p^2}$-rational j-invariant, the SIDH protocol remains unchanged when Alice and Bob are free to work in both sets of twists.

This framework lifts the restrictions on the shapes of the underlying prime fields originally imposed by Jao and De Feo, and allows a range of new options for instantiating isogeny-based public key cryptography. These include alternatives that exploit Mersenne and Montgomery-friendly primes, as well as the possibility of halving the size of the primes in the Jao-De Feo construction at no known loss of asymptotic security. For a given target security level, the resulting public keys are smaller than the public keys of all of the key encapsulation schemes currently under consideration in the NIST post-quantum standardisation effort.

The best known attacks against the instantiations proposed in this paper are the classical path finding algorithm due to Delfs and Galbraith and its quantum adapation due to Biasse, Jao and Sankar; these run in respective time $O(p^{1/2})$ and $O(p^{1/4})$, and are essentially memory-free. The upshot is that removing the big-O's and obtaining concrete security estimates is a matter of costing the circuits needed to implement the corresponding isogeny. In contrast to other post-quantum proposals, this makes the security analysis of B-SIDH rather straightforward.

1. Introduction

The best known attacks against Jao and De Feo's SIDH protocol [22] try to recover either Alice's secret 2^m-isogeny $\phi_A\colon E_0 \to E_A$, or Bob's secret 3^n-isogeny $\phi_B\colon E_0 \to E_B$, and both of these problems are instances of the *supersingular isogeny problem*: given a finite field K and two supersingular elliptic curves E, E' defined over K such that $\#E = \#E'$, compute an isogeny $\phi\colon E \to E'$. For the cases of interest where $K = \mathbb{F}_{p^2}$ and p is a large prime, the best known classical algorithm for solving the supersingular isogeny problem is the Delfs-Galbraith algorithm [14], which requires $O(p^{1/2})$ isogeny operations to find a collision (of walks from E and E') in the graph of size $O(p)$. However, the special isogenies computed in SIDH above give rise to appreciably easier instances of the supersingular isogeny problem; they are of a fixed, known degree close to $p^{1/2}$, and this allows for a classical meet-in-the-middle attack that, asymptotically, requires only $O(p^{1/4})$ isogeny operations [22, §5]. Roughly speaking, the difference between the difficulty of the isogeny problems that arise in SIDH and that of the general supersingular isogeny problem is due to the fact that Alice and Bob only take about half as many steps as the diameters of each of their graphs. In other words, the number of possible destination nodes for the secret walks of Alice and Bob is close to the square root of the total number of nodes in the graph.

Jao and De Feo chose primes of the form $p = 2^m 3^n - 1$ and half-length walks so that Alice and Bob can both compute their isogenies using arithmetic in $\mathbb{F}_{p^2}$; they represent each isomorphism class by a supersingular elliptic curve $E/\mathbb{F}_{p^2}$ with group order $\#E(\mathbb{F}_{p^2}) = (p+1)^2 = (2^m 3^n)^2$, which facilitates a full $\mathbb{F}_{p^2}$-rational 2^m-torsion and full $\mathbb{F}_{p^2}$-rational 3^n-torsion. When all of the subgroups of order 2^m and 3^n are $\mathbb{F}_{p^2}$-rational, so are the corresponding isogeny computations.

A first observation that sets the scene for this work is that there are actually two choices of group orders for supersingular curves over $\mathbb{F}_{p^2}$: those whose group orders are $(p+1)^2$, and those whose group orders are $(p-1)^2$. Moreover, this choice can be made at every node in the supersingular isogeny graph: although curves from these two sets are not isomorphic (or even isogenous!) to one another over $\mathbb{F}_{p^2}$, they do become isomorphic over $\mathbb{F}_{p^4}$, and therefore share the same j-invariant in $\mathbb{F}_{p^2}$ [36, Proposition III.1.4]. Indeed, for any curve whose group order is $(p+1)^2$, its *quadratic twist* over $\mathbb{F}_{p^2}$ has group order $(p-1)^2$.

The main point of this paper is to exploit the fact that the SIDH protocol does not have to restrict to working in one of the two sets of quadratic twists: it can stay in $\mathbb{F}_{p^2}$ while working in *both* the $(p+1)$-torsion *and* the $(p-1)$-torsion. Moreover, Alice and Bob can work in the torsion corresponding to opposite sets of quadratic twists with no change to the protocol. Optimised Montgomery arithmetic [29] in the SIDH setting only needs the x-coordinates of points [22] and the A coefficient of the curve [11], and as such is entirely *twist-agnostic*; in other words, the twisting morphism (which only alters y-coordinates and the B coefficient) leaves x-coordinates and A coefficients unchanged, so the lifting to $\mathbb{F}_{p^4}$ described above becomes a mere theoretical technicality that is not visible in cryptographic implementations – see Section 3.

The price to pay for working with both twists is that at least one of Alice or Bob must now perform walks comprised of steps in multiple ℓ-isogeny graphs, i.e. switching between multiple values of ℓ. This changes the underlying hardness assumption for one or both parties, but (as is discussed in Section 4) there is no known reason to believe that switching between many ℓ's makes the resulting SIDH problems any easier, so long as the number of destination nodes remain roughly the same size as in the Jao-De Feo instantiation.

Allowing torsion from both sets of twists unlocks a number of new options and trade-offs for isogeny-based public key cryptography; many examples are given in Section 5 to illustrate these possibilities. At a high level, these options fall into two categories: the first is where Alice gets to computes significantly faster 2^m isogenies (than in existing SIDH/SIKE implementations) at the expense of a heavy slowdown on Bob's side; the second, and perhaps the more interesting, is the possibility of halving the sizes of the underlying fields at no known loss of asymptotic security. Furthermore, this possibility gives rise to the number of secret walks (i.e. possible destination nodes) for both Alice and Bob being very close to the total number of nodes in the graph.

Concrete instantiations of smaller primes are put forward in §5.2. For example, B-SIDHp247 uses a 247-bit prime to achieve roughly the same conjectured security as the 434-bit SIKE prime to target NIST's security category 1 [21]. The public keys for B-SIDHp247 are 186 bytes, which are a little over half the size of the 330-byte uncompressed public keys of SIKEp434, and are still smaller than the 196-byte keys that are obtained in SIKEp434 when compression is enabled.

1.1. **Naming.** The instantiation proposed in this paper is dubbed B-SIDH[1] in order to distinguish it from the original Jao-De Feo SIDH instantiation, and to avoid muddying the waters in the case that future cryptanalysis weakens any variants described herein. Although switching between multiple ℓ-isogeny graphs during a secret isogeny computation does not decrease security in any known way, it may turn out that using torsion with many prime factors is a bad idea, or that decreasing p relative to the degrees of the secret isogenies is a bad idea. Of course, it may also turn out that the one (or both) of the converse statements is true, but in any case it must be emphasised that the instantiations proposed in this paper rely on *different security assumptions* than SIDH and SIKE – see Section 4.

1.2. **Performance vs. SIDH.** There are no performance claims made in this paper, except in the scenarios where Alice's performance will clearly be improved (over her performance in the SIDH/SIKE setting at a comparable security level) thanks to a faster underlying prime, but where it should be reiterated that Bob will almost always suffer a collossal slowdown. The main takeaway of this paper is that the primes and the public keys in the optimal scenario of §5.2 are significantly smaller than the SIDH/SIKE counterparts. Moreover, the public keys in §5.2 will remain smaller,

[1]Pronounced "B-side", in reference to the analogy between the set of supersingular curves of cardinality $(p-1)^2$ and the less popular, sometimes forgotten 'flip-side' of a record. Additionally, unlike the original set of supersingular curves of cardinality $(p+1)^2$, writing a supersingular curve from the B-side in Montgomery form cannot be done without specifying a non-trivial value of the B coefficient– see Section 3.

even when compression techniques [2, 10, 43, 30] are applied to the SIDH and SIKE public keys. If the ECC+SIDH/SIKE hybrid is used as in [11], these gaps will widen further.

In order to make the performance of the proposed approach competitive with that of SIDH/SIKE, the main research obstacles that arise are (i) finding faster methods of computing ℓ-isogenies for the sizes of ℓ that arise in Section 5, and (ii) finding even *smoother* instances than those presented in Table 1. Both of these avenues are left open for future work.

1.3. **Related work.** A few days after a preprint of this paper went online, Matsuo sent us his non-peer-reviewed Japenese article [27] from March 2019 that had previously proposed the idea of working in both quadratic twists simultaneously. While Matsuo is therefore the first to propose working in both sets of torsion, it must be emphasised that his execution of the idea is very different from that in this paper. In particular, Matsuo did not lift the restriction of Alice and Bob computing their respective 2^m and 3^n isogenies, and his search for primes p such that $2^m \mid p+1$ and $3^n \mid p-1$ (or vice versa) forces huge cofactors which produces primes that are, for the most part, either the same size or are larger than their original SIDH counterparts. A crucial difference in this work is allowing at least one of the two parties to compute secret isogenies whose composite degrees have many prime factors, which gives way to a range of new possibilities.

2. Twist-agnostic SIDH

The parameter that governs the security of Jao and De Feo's supersingular isogeny Diffie-Hellman (SIDH) protocol is the large prime p. As soon as p is chosen, a set of roughly $\lfloor p/12 \rfloor$ elements is defined: these are the entire set of supersingular j-invariants over $\overline{\mathbb{F}}_p$, and they are the nodes on the graphs that Alice and Bob walk on during the protocol. Alice and Bob share this set of nodes, but their graphs have different edges that depend on the degrees of their secret isogenies. Following [22], for any prime $\ell \nmid p$, there are $\ell+1$ isogenies (counting multiplicities, and up to isomorphism) of degree ℓ that eminate from a given supersingular isomorphism class. Moreover, Pizer [32, 33] showed that this gives rise to a connected $(\ell+1)$-regular multigraph that satisfies the Ramanujan property and thus has optimal expansion properties.

2.1. **Rational** $(p+1)$**-torsion.** The prime p also governs the efficiency of SIDH. Alice and Bob both compute prime power isogenies whose degrees are of the form ℓ^e, where the only restrictions are that their individual values of ℓ are coprime and that the size(s) of ℓ^e is large enough that recovering the secret isogeny walks is hard. In theory, Alice and Bob could choose any value of ℓ they like, but it is more efficient if the ℓ^e-torsion is defined over $\mathbb{F}_{p^2}$. Observing that the smallest primes ℓ give rise to the most efficient ℓ^e-isogenies, Jao and De Feo construct the prime p to guarantee this rationality condition by setting $p = f \cdot 2^m 3^n - 1$ (allowing for a small *cofactor* f), and representing nodes in the graph by elliptic curves $E/\mathbb{F}_{p^2}$ with

$$E(\mathbb{F}_{p^2}) \;\cong\; \mathbb{Z}_{p+1} \times \mathbb{Z}_{p+1}. \tag{1}$$

For any $r \in \mathbb{Z}$ with $r \mid p+1$, the entire r-torsion $E[r] \cong \mathbb{Z}_r \times \mathbb{Z}_r$ is then contained in $E(\mathbb{F}_{p^2})$. With p chosen as above, it follows that the full 2^m-torsion $E[2^m] \cong \mathbb{Z}_{2^m} \times \mathbb{Z}_{2^m}$, and the full 3^n-torsion $E[3^n] \cong \mathbb{Z}_{3^n} \times \mathbb{Z}_{3^n}$, are both $\mathbb{F}_{p^2}$-rational. Since every (seperable) isogeny $\phi\colon E \to E'$ of degree d is in one-to-one correspondence with a kernel subgroup of order d [36, Proposition III.4.12], and each such isogeny is computed using rational functions of the input curve and the given kernel subgroup [41], it follows that if both of these inputs are $\mathbb{F}_{p^2}$-rational, then so is the isogeny computation.

2.2. **SIDH.** With $p = f \cdot 2^m 3^n - 1$ as above, the SIDH protocol specifies the following public parameters: a starting supersingular curve $E_0/\mathbb{F}_{p^2}$, a basis $\{P_A, Q_A\}$ for $E[2^m] \cong \mathbb{Z}_{2^m} \times \mathbb{Z}_{2^m}$, and a basis $\{P_B, Q_B\}$ for $E[3^n] \cong \mathbb{Z}_{3^n} \times \mathbb{Z}_{3^n}$. To generate her public key, Alice chooses two secret integers $(\alpha_A, \beta_A) \in \mathbb{Z}_{2^m} \times \mathbb{Z}_{2^m}$ such that her secret point $S_A = [\alpha_A]P_A + [\beta_A]Q_A$ is of order 2^m. She then composes m 2-isogenies to give her secret 2^m-isogeny $\phi_A\colon E_0 \to E_A$, where $E_A = E_0/\langle S_A \rangle$. Along the way, she moves the basis points P_B and Q_B through the isogeny computation, eventually obtaining their images under ϕ_A. Her public key is then $\mathrm{PK}_A = (\,E_A\,, \phi_A(P_B)\,, \phi_A(Q_B))$. On Bob's side, he chooses $(\alpha_B, \beta_B) \in \mathbb{Z}_{3^n} \times \mathbb{Z}_{3^n}$, computes his secret point $S_B = [\alpha_B]P_B + [\beta_B]Q_B$, and then uses it to compute his secret 3^n-isogeny $\phi_B\colon E_0 \to E_B$ (via n consecutive 3-isogenies), such that $E_B = E_0/\langle S_B\rangle$. His public key is $\mathrm{PK}_B = (\, E_B\,, \phi_B(P_A)\,,\, \phi_B(Q_A))\,$.

Upon receiving PK_B, Alice uses her secret integers to compute a new secret point $S'_A = [\alpha_A]\phi_B(P_A) + [\beta_A]\phi_B(Q_A)$ of order 2^m on E_B, and then uses it to compute the 2^m-isogeny $\phi'_A\colon E_B \to E_B/\langle S'_A\rangle$. Bob uses his secret integers and PK_A to compute the point $S'_B = [\alpha_B]\phi_A(P_B) + [\beta_A]\phi_A(Q_B)$ of order 3^n on E_A, and then uses it to compute the 3^n-isogeny $\phi'_B\colon E_A \to E_A/\langle S'_B\rangle$. Both parties then compute the same shared secret as the j-invariant of their respective image curves $E_B/\langle S'_A\rangle$ and $E_A/\langle S'_B\rangle$, since $E_B/\langle S'_A\rangle \cong E_A/\langle S'_B\rangle$ [22].

2.3. **Twist-agnostic isogenies.** Jao and De Feo exploited the fact that all of the arithmetic in the above computations can be performed on the Kummer line of the associated curves, i.e. in $E/\{\pm 1\}$ rather than E, and furthermore that this arithmetic is particularly efficient if the curves are in Montgomery form [29]

$$E_{(A,B)}\colon By^2 = x^3 + Ax^2 + x.$$

Henceforth, $E_{(A,B)}$ or E will be used instead of $E_{(A,B)}/\{\pm 1\}$ or $E/\{\pm 1\}$ for simplicity, and unless explicitly stated, y-coordinates will be ignored (using '—'). Furthermore, the B coefficients of Montgomery curves can also be ignored in the SIDH framework [11]; they are merely used to specify which quadratic twist we are working on and are not needed in optimised explicit formulas. In other words, optimised explicit formulas for Montgomery arithmetic ignore B and y and work irrespective of quadratic twist.

Following [9], for any point $P \in E_{(A,—)}$ of prime order ℓ, the ℓ-isogeny

$$\phi\colon E_{(A,—)} \to E_{(A',—)}\,, \qquad (x, —) \mapsto (x', —)$$

with $\ker(\phi) = \langle P\rangle$ can be computed in $O(\ell)$ field operations. Isogenies of composite degree $L = \prod_{i=1}^{k} \ell_i^{e_i}$ can be computed as the composition of e_1 isogenies of degree ℓ_1, followed by e_2 isogenies of degree ℓ_2, and so on. It follows that the complexity a general L-isogeny is closely tied to both the size of L and the size of L's largest prime factor.

3. Using torsion from the quadratic twists

Let $E/\mathbb{F}_{p^n}$ be an elliptic curve, let t_n be the trace of the p^n-power Frobenius endomorphism, and recall that (i) E is supersingular if and only if t_n is a multiple of p [36, Exercise V.5.10(a)], and that (ii) $\#E(\mathbb{F}_{p^n}) = p^n + 1 - t_n$ with $|t_n| \le 2\sqrt{p^n}$ [36, Theorem V.1.1]. When $n = 1$, there is only one possible value of t_1 that is a multiple of p such that $|t_1| \le 2\sqrt{p}$, i.e. $t_1 = 0$, and thus it follows that $E/\mathbb{F}_p$ is supersingular if and only if $\#E(\mathbb{F}_p) = p + 1$. In other words, there is only one possible group order for supersingular elliptic curves over $\mathbb{F}_p$.

The first observation that sets the scene for this work is that there are actually two possible values for t_2 that correspond to $E/\mathbb{F}_{p^2}$ being supersingular: $t_2 = -2p$ and $t_2 = 2p$ both satisfy (i) and (ii). All known instantiations of SIDH and SIKE fall into the former case by default. They define a starting supersingular curve $E_0/\mathbb{F}_p$ and lift to work in $E_0(\mathbb{F}_{p^2})$; since $E_0(\mathbb{F}_p) \mid E_0(\mathbb{F}_{p^2})$ and $\#E_0(\mathbb{F}_p) = p+1$, it must be that $\#E_0(\mathbb{F}_{p^2}) = p^2+1+2p = (p+1)^2$ and hence that $t_2 = -2p$.

Upon starting on a curve with $t_2 = -2p$, a choice has been made between the two possibilities for t_2; two elliptic curves are $\mathbb{F}_{p^2}$-isogenous if and only if they have the same group order over $\mathbb{F}_{p^2}$ [39, Theorem 1(c)], so computing $\mathbb{F}_{p^2}$-rational isogeny walks means walking on curves with the same number of points as $E_0/\mathbb{F}_{p^2}$. However, these two possibilities for t_2 correspond to sets of curves that are the *quadratic twists* of one another, meaning that they not only become isogenous over $\mathbb{F}_{p^4}$, they become isomorphic over $\mathbb{F}_{p^4}$. Moreover, as we saw in §2.3, optimised isogeny arithmetic works correctly independently of the quadratic twist, so the explicit formulas that are used on the curves with $t_2 = -2p$ can also be used to work on the curves with $t_2 = 2p$.

It is crucial to note that even though two quadratic twists are not isomorphic over $\mathbb{F}_{p^2}$, they will still have the same j-invariant in $\mathbb{F}_{p^2}$ [36, Proposition 1.4(b)]. Put another way, every node in the supersingular isogeny graph can actually be represented by two different $\mathbb{F}_{p^2}$-isomorphism classes: those with $t_2 = -2p$ and the same group structure as $E/\mathbb{F}_{p^2}$ in (1), or those with $t_2 = 2p$ and with group structure

$$E^t(\mathbb{F}_{p^2}) \;\cong\; \mathbb{Z}_{p-1} \times \mathbb{Z}_{p-1}.$$

Every such supersingular curve with group structure $\mathbb{Z}_{p-1} \times \mathbb{Z}_{p-1}$ is the *quadratic twist* of a supersingular curve with group structure $\mathbb{Z}_{p+1} \times \mathbb{Z}_{p+1}$, and vice versa. Moreover, in the same way that any factor r of $p+1$ gave rise to a full rational r-torsion in $E(\mathbb{F}_{p^2})$, any factor s of $p-1$ gives rise to a full rational s-torsion in $E^t(\mathbb{F}_{p^2})$.

For Alice and Bob to freely work with points coming from the $(p+1)$-torsion and the $(p-1)$-torsion, it appears that the entire protocol must be lifted to $\mathbb{F}_{p^4}$. While this is technically true, the lifting will ultimately not be visible in an optimised implementation [2]. The point and isogeny formulas ignore the y-coordinates of points and the B coefficients of Montgomery curves, and this is where all the twisting arithmetic happens. The upshot is that while the protocol will be lifted to $\mathbb{F}_{p^4}$, where $E(\mathbb{F}_{p^4}) \;\cong\; E^t(\mathbb{F}_{p^4}) \;\cong\; \mathbb{Z}_{p^2-1} \times \mathbb{Z}_{p^2-1}$, Alice and Bob are still in a position to work entirely in $\mathbb{F}_{p^2}$ as usual. They can then choose a secret kernel point whose order divides $p+1$, or whose order divides $p-1$, or (more generally) whose order divides the product p^2-1.

To make this concrete, let B be a square in $\mathbb{F}_{p^2}$, let γ be a non-square in $\mathbb{F}_{p^2}$, take $\mathbb{F}_{p^4} = \mathbb{F}_{p^2}(\delta)$ with $\delta^2 = \gamma$, and write

$$E_{A,B}\colon By^2 = x^3 + Ax^2 + x \qquad \text{and} \qquad E^t_{A,\gamma B}\colon \gamma By^2 = x^3 + Ax^2 + x$$

as models[3] for $E/\mathbb{F}_{p^2}$ and $E^t/\mathbb{F}_{p^2}$. The map

$$\sigma\colon E_{A,\gamma B}(\mathbb{F}_{p^4}) \to E_{A,B}(\mathbb{F}_{p^4}), \qquad (x,y) \mapsto (x, \delta y) \tag{2}$$

is a group isomorphism that leaves x-coordinates unchanged.

Write $f(x) = x^3 + Ax^2 + x$. For any $u \in \mathbb{F}_{p^2}$, either (i) $f(u)$ is a square in $\mathbb{F}_{p^2}$, in which case $(u, \sqrt{f(u)/B})$ is a point in $E_{A,B}(\mathbb{F}_{p^2})$, (ii) $f(u)$ is a non-square in $\mathbb{F}_{p^2}$, in which case $f(u)/(\gamma B)$ is a square, and $(u, \sqrt{f(u)/(\gamma B)})$ is a point in $E_{A,\gamma B}(\mathbb{F}_{p^2})$, or (iii) $f(u) = 0$, in which case $(u, 0)$ is one of the three 2-torsion points (on both $E_{A,B}$ and $E_{A,uB}$).

Let $P_1 = (u_1, —)$ be a point corresponding to case (i), let $P_2 = (u_2, —)$ be a point corresponding to case (ii), and suppose $\phi_1\colon E_{A,B} \to E_{A,B}/\langle P_1\rangle$ and $\phi_2\colon E_{A,\gamma B} \to E_{A,\gamma B}/\langle P_2\rangle$. It does not make sense to evaluate ϕ_1 at P_2 or ϕ_2 at P_1 (these points do not even lie on $\mathbb{F}_{p^2}$-isogenous curves, let alone the same curve), but this is fixed by lifting to $\mathbb{F}_{p^4}$ and precomposing with the twisting morphisms. Setting $\phi_1' = (\phi_1 \circ \sigma)$ and $\phi_2' = (\phi_2 \circ \sigma^{-1})$ gives the isogenies $\phi_1' : E_{A,\gamma B} \to E_{A,\gamma B}/\langle \sigma(P_2)\rangle$ and $\phi_2' : E_{A,B} \to E_{A,B}/\langle \sigma^{-1}(P_1)\rangle$, which are well-defined over $\mathbb{F}_{p^4}$.

The key observation from (2) is that $\sigma\colon (x, —) \mapsto (x, —)$ and $\sigma^{-1} : (x, —) \mapsto (x, —)$ behave like the identity map when working on the corresponding Kummer lines, so the twisting morphisms can simply be ignored in the implementation. Thus, Alice can take her secret points from the $(p+1)$-torsion of $E_{A,B}(\mathbb{F}_{p^2})$ and Bob can take his secret points from the $(p-1)$-torsion of $E_{A,\gamma B}$, and the implementation of the SIDH protocol can otherwise remain unchanged.

3.1. **B-SIDH in a nutshell.** Henceforth, for a given prime p, M and N will be used to denote the two coprime degrees of Alice and Bob's secret isogenies (e.g. in the traditional setup with $p = 2^m 3^n - 1$ described above, we have $M = 2^m$ and $N = 3^n$). Alice's degree M will always be defined such that $M \mid p+1$, and the most common case for Bob will be when $N \mid p-1$. In some instances, however, it can be preferable to comprise N as a product of factors from both sides, i.e. taking $N \mid p^2 - 1$ where $N \nmid p-1$ and $N \nmid p+1$; in such cases, a caveat applies – see §3.2 below.

Since M and N must be coprime, the even one will always be chosen according to whichever of $p+1$ and $p-1$ is the multiple of 4; otherwise, the remaining factors of $p+1$ and $p-1$ are necessarily coprime. The efficacy of the construction in this paper is closely tied to the *smoothness* of M and N (see §2.3), so obtaining B-SIDH-friendly parameters boils down to searching for primes p such

[2] This is reminiscent of Bernstein's twist-agnostic Curve25519 construction. He also uses a quadratic extension field in the specification of the Curve25519 function [3, Theorem 2.1], but this extension is a technicality that is not seen in the implementation.

[3] The idea works analogously for more general (i.e. short Weierstrass) elliptic curves, but all of the instantiations discussed in this paper allow for Montgomery form.

that $p+1$ and $p-1$ both contain factors that are large enough to reach a target security level, but smooth enough to be efficiently computable.

3.2. **Bob on both sides: a caveat.** As mentioned above, there will be some scenarios where Bob is better off computing isogenies of order $N = N_1N_2$, where $N \nmid p-1$ but where $N_1 \mid p+1$ and $N_2 \mid p-1$. In this case, general points in $E_{A,B}[N]$ no longer have their x-coordinate in $\mathbb{F}_{p^2}$, but rather in $\mathbb{F}_{p^4}$, and performing arithmetic in $\mathbb{F}_{p^4}$ would hamper the efficiency of the isogeny algorithms significantly. One way to approach this scenario is to instead have Bob use two bases $\langle P_1, Q_1\rangle = E_{A,B}[N_1]$ and $\langle P_2, Q_2\rangle = E_{A,\gamma B}[N_2]$, which can both be defined such that all four x-coordinates are in $\mathbb{F}_{p^2}$. His secret keys are then of the form $(s_1, s_2) \in [0, N_1) \times [0, N_2)$, which generate the secret kernels $S_1 = P_1 + [s_1]Q_1$ and $S_2 = P_2 + [s_2]Q_2$. Bob can compute $\phi_1 \colon E_0 \to E_0/\langle S_1\rangle$ and then $\phi_2 \colon E_0/\langle S_1\rangle \to (E_0/\langle S_1\rangle)/\langle \phi_1(S_2)\rangle$, which corresponds to the secret isogeny $\phi_B = (\phi_2 \circ \phi_1)$; his public key is then $(E_B, P'_A, Q'_A) = (\phi_B(E_0), \phi_B(P_A), \phi_B(Q_A))$, which is the same size as usual. On the other side, Alice's public keys must include the images of all four of Bob's basis points under her secret isogeny, so they become significantly larger (if a static-ephemeral version of Diffie-Hellman à la SIKE [21] is used, then the setup would likely be arranged to make the static key the larger key). Computing these extra image points also incurs some additional overhead, but this would still be faster than working with two basis points that are defined over $\mathbb{F}_{p^4}$.

3.3. **Handling large ℓ-degree isogenies.** Recall from §2.3 that, in contrast to the multiplication-by-ℓ map which can be computed in $\log(\ell)$ field operations, the computation of a prime degree ℓ-isogeny requires $O(\ell)$ field operations. As the sizes of ℓ that are encountered in this paper are significantly larger than those in previous works, it is important to look for ways that such isogenies can be sped up in practice.

3.3.1. *Parallelisation.* Let P be a point of order $\ell = 2d+1$. The algorithm in [9] requires the first d multiples $\{[i]P\}_{1\le i\le d}$ of the input point, which is what makes ℓ-isogeny computations become rather expensive for large ℓ. However, this process parallelises almost perfectly: for t processors, $\lceil t/2\rceil$ steps of the Montgomery ladder are used to compute $[i]P$ for $1 \le i \le t$. The i-th processor can then compute $[i+jt]P$ as the differential sum of $[i+(j-1)t]P$, $[t]P$, and $[i+(j-2)t]P$ for $1 \le j \le \lceil d/t\rceil$. After the initial phase that assigns the three values to each processor, no communication is required between the processors until the end, where the subproducts (which were independently accumulated in the same manner as [9, §5]) can all be collected and multiplied together. In the case of computing image points, then one final squaring and one final multiplication are used to finish the routine [9, Theorem 1]; in the case of computing image curves, then $\log(\ell)$ final multiplications and squarings are required [28]. Note that this parallelisation can be exploited across any of the prime degree isogenies that are large enough to make it worthwhile.

3.3.2. *Precomputation.* Assume Bob is tasked with large prime degree isogenies and he is the one generating ephemeral public keys. The runtime of his public key generation procedure can be improved if storage permits a significant offline precomputation. For example, if his largest prime-degree isogeny is an ℓ-isogeny, he could precompute all of the $\ell+1$ possible image curve/point triples (see §2.2), and at runtime he could simply select the triple corresponding to his secret key.

4. Security analysis

There are two main changes to the usual computational isogeny problems underlying SIDH and SIKE [15, Problems 5.1–5.4] that are implicit in this paper. The first is that the isogeny walks now use multiple values of ℓ; the vertex set of a given graph stays fixed, but the edges now change between successive steps. The second is that the walks are no longer *half-length* (i.e. around half the bitlength of p); lowering the size of the primes relative to the length of the walks means that other avenues of attack become relevant with respect to the usual meet-in-the-middle attacks. This section studies the implications of these changes with respect to known attacks from the literature.

4.1. **Multiple edge sets.** Based on current knowledge, there is no reason to believe that a walk consisting of many different prime degree isogenies makes the underlying problem appreciably easier than that of a walk in a fixed ℓ-isogeny graph, provided the number of possible destination nodes is around the same size. When computing L-isogenies with $L = \prod \ell_i^{e_i}$, the number of cyclic subgroups of order L inside any given group $E(\mathbb{F}_{p^2})$ is $\prod(\ell_i+1)\ell_i^{e_i-1}$, and so long as this is around the same size as $(\ell+1)\ell^{e-1}$, the difficulty of recovering an L-isogeny appears to be no easier than that of recovering an ℓ^e-isogeny. The generalisation of the problems underlying SIDH to isogenies of multiple degrees has already been considered in prior works (e.g. [31], [18, §2.3], and [7]), where the same conclusion was drawn (or the same assumption was made).

4.2. **Security of non-commutative vs. commutative schemes.** There are currently two main umbrellas of isogeny-based public-key cryptography under public scrutiny: those like SIDH [22] and SIKE [21] where the curves involved have non-commutative endomorphism rings, and those like CRS [13, 35] and CSIDH [6] where the associated endomorphism rings are commutative. It is important to note that, while there are similarities between the instantiations herein and CSIDH (like the use of many different prime isogeny degrees in the same secret computation), this paper falls entirely under the non-commutative umbrella. This means B-SIDH inherits two security virtues from SIDH: the first is that it is seemingly immune to Kuperberg's algorithm [24], meaning that the best known quantum algorithms are exponential (see §4.4); the second is that it lends itself to regular algorithms and therefore more simple constant-time implementations. On the other hand, it inherits the same drawback as SIDH of being susceptible to active attacks [17], so requires the same transformations that were used in the SIKE proposal – see [21].

4.3. **Classical cryptanalysis.** When $L = \prod \ell_i^{e_i} \approx p^{1/2}$, as in the original SIDH proposal, the *meet-in-the-middle* or *claw-finding* algorithms [15, §5.3] stand alone as the best known attacks against SIDH and SIKE. However, the most interesting instantiations proposed in this paper have $L \gg p^{1/2}$, and as L tends towards p, algorithms other than the meet-in-the-middle attacks become relevant. In what follows it will be assumed that $L \approx p$, since this is the extreme case where the alternative attack avenues are most relevant. The underlying problem is to find the isogeny $\phi\colon E_1 \to E_2$ of degree L, where $E_1/\mathbb{F}_{p^2}$ and $E_2/\mathbb{F}_{p^2}$ are supersingular.

4.3.1. *Claw-finding algorithms.* Let $L_1 \approx L_2 \approx p^{1/2}$ with $L_1L_2 = L$. The claw-finding algorithm cited by Jao and De Feo [22, §5.2] uses $O(L_1)$ time to compute a table of all of the curves L_1-isogenous to E_1, and stores them using $O(L_1)$ memory. It then proceeds by trying one L_2-isogeny at a time, this time emanating from E_2, until a match is found in the table and the problem is solved; this stage requires $O(L_2)$ time and essentially no memory. It follows that the claw-finding algorithm runs in $O(p^{1/2})$ time and requires $O(p^{1/2})$ memory.

Adj, Cervantes-Vázquez, Chi-Domínguez, Menezes and Rodríguez-Henríquez [1] argued that the van Oorschot-Wiener (vOW) parallel collision finding algorithm [40] has a lower overall cost for finding ϕ, and thus should be used to assess the security of SIDH and SIKE. Their implementation confirmed that the original vOW runtime analysis [40] is sharp in the context of finding the isogeny ϕ. If w is the number of entries that can be stored in the table above, m is the number of processors running in parallel, and t is the time taken to compute L_1 and L_2 isogenies, then the vOW algorithm finds ϕ in expected runtime $T = \frac{2.5}{m} \cdot \left(\frac{p^{3/4}}{w^{1/2}}\right) \cdot t$. Adj et al. conclude that $w > 2^{80}$ is infeasible, so conduct their analysis by setting $w = 2^{80}$. With this choice of w, it helps to point out that for $p = 2^{160}$, the runtime of vOW (on one processor) is $T = 2.5 \cdot t \cdot p^{1/2}$; thus, when $p \gg 2^{160}$, the vOW runtime is $T \gg p^{1/2}$.

4.3.2. *Random walk algorithms for any path.* There are two styles of applicable random walk algorithms that can be used to solve the general supersingular isogeny problem: both Pollard rho [34] and Delfs-Galbraith [14] find *some* path between E_1 and E_2. The former finds an isogeny between E_1 and E_2 by taking two pseudo-random walks in the graph of size $O(p)$; the number of steps required until these two walks collide is $O(p^{1/2})$ by the birthday paradox. The latter algorithm, which is preferred in practice (see [14, §4] or [4]), uses two self-avoiding random walks to find paths from each curve to two subfield curves, $\tilde{E}_1/\mathbb{F}_p$ and $\tilde{E}_2/\mathbb{F}_p$, and then connects these two subfield curves. Since there are $O(p^{1/2})$ subfield curves in the graph of size $O(p)$, the first step requires $O(p^{1/2})$ steps, and since connecting the two subfield curves requires $O(p^{1/4})$ steps [14], the entire algorithm takes $O(p^{1/2})$ steps to find an isogeny connecting E_1 and E_2. Like vOW, the Delfs-Galbraith algorithm parallelises perfectly, but unlike vOW, it does not have large storage requirements.

Both of these algorithms are likely to terminate with a path that is not the secret path corresponding to ϕ. However, since E_1 is typically a special curve with a known endomorphism ring $\mathrm{End}(E_1)$, it is prudent to assume that this can be used to modify the path into the correct one via the techniques discussed at length in [17, §4].

4.4. **Quantum cryptanalysis.** The best known quantum algorithm for solving SIDH and SIKE instances is, asymptotically, Tani's algorithm [38]. Roughly speaking, as $p \to \infty$, Tani's algorithm solves the claw-finding problem for secret isogenies of degree $O(p^{1/2})$ in time $O(p^{1/6})$ on a quantum computer. Translating to the setting of isogenies of degree $L \approx p$, this would give an $O(p^{1/3})$ quantum claw-finding algorithm; note that recent work of Jaques and Schanck [23] shows that (even under the assumption of a large amount of quantum resources) the concrete complexity of Tani's algorithm is much closer to the classical claw-finding complexity. Nevertheless, when $L \approx p$, Tani's algorithm is no longer than superior algorithm for solving the corresponding isogeny problem. In [4], Biasse, Jao and Sankar give a quantum algorithm for the general supersingular isogeny problem (in characteristic p) that runs in time $O(p^{1/4})$. Their algorithm is essentially the Delfs-Galbraith algorithm (from above) ported to the quantum setting; they use Grover's algorithm [19] to get a quadratic speedup from $O(p^{1/2})$ to $O(p^{1/4})$ on the phase that finds the two supersingular subfield curves $\tilde{E}_1/\mathbb{F}_p$ and $\tilde{E}_2/\mathbb{F}_p$, and then develop a subexponential algorithm (based on the Childs-Jao-Soukharev subexponential algorithm [8] for the ordinary case) to connect the subfield path. The memory requirements of this algorithm are small; Biasse, Jao and Sankar define a set of N isogenies of degree 3^λ, where $\lambda \in O(\log(p))$ is chosen large enough so that this set contains a walk that passes through a subfield curve with probability $1/2$. As long as there are enough (i.e. $O(\log(p))$) qubits to encode such a path, then this algorithm succeeds with probability $1/4$ [4, Proposition 2].

As in the classical algorithms, since $\mathrm{End}(E_1)$ is typically known, the path obtained by the above process can presumably be modified into the path corresponding to ϕ at no additional asymptotic cost.

4.5. **Security summary.** When $\phi\colon E_1 \to E_2$ is an isogeny between two supersingular curves $E_1/\mathbb{F}_{p^2}$ and $E_2/\mathbb{F}_{p^2}$ of degree $L = \prod_{i=1}^{k} \ell_i^{e_i} \approx p$, the best known classical algorithm for finding ϕ is the Delfs-Galbraith algorithm [14]; it runs in $O(p^{1/2})$ time and (unlike claw-finding or vOW) does not have large storage requirements. Applying Grover's speedup to the Delfs-Galbraith algorithm also gives the best known quantum algorithm [4]; it requires $O(\log(p))$ qubits, run in time $O(p^{1/4})$, and does not have large storage requirements. In the classical case, Delfs-Galbraith parallelises perfectly, where as Grover's algorithm is well-known to give a $\sqrt{m}$ speedup when parallelised across m quantum processors [42].

5. Searching for friendly instances

This section presents a variety of example primes for which the approach in this paper becomes interesting in practice.

5.1. **Fast fields: accelerating Alice, burdening Bob.** Many interesting alternatives come from primes of the form

$$p = 2^m \cdot c - 1, \tag{3}$$

which allow Alice to compute 2^m-isogenies just like she would in SIDH. However, unlike the primes in SIDH where $c = 3^n \approx 2^m$, the values of c that are of interest here are when c is either chosen to facilitate faster field arithmetic in $\mathbb{F}_{p^2}$, is much smaller than 2^m so that p is smaller than usual, or both. Here Alice's computations will benefit from the faster field arithmetic, but Bob's computations become significantly slower due to his isogenies no longer being 3^n-isogenies, but rather ($\prod \ell_i^{e_i}$)-isogenies (recall from §2.3 that the cost in this case is dependent on the size of the largest prime ℓ_i). Depending on the efficacy of the methods in §3.3, in almost all such cases the factor slowdown incurred on Bob's side will be much worse than the factor speedup enjoyed by Alice, meaning that the runtime of one protocol instance will be significantly slower in general. However, there are real-world scenarios where such a trade-off would be welcomed. One such scenario is in TLS, where servers are oftentimes performing orders of magnitude more runs of the protocol than an individual client is; here slowdowns on the client side could be tolerated (or even unnoticed) to afford a speedup to the server. An example of the opposite scenario, i.e. when the priority becomes the client's performance, is in the arena of lightweight cryptography (e.g. IoT); here it is often the case that resource-constrained devices are communicating with a relatively unconstrained sever.

5.1.1. *Mersenne primes.* Putting $c = 1$ into (3) yields Mersenne primes, for which only $m \in \{127, 521\}$ are of interest in this paper. With $m = 521$, write the factorisation $p - 1 = 2^{521} - 2 = 2 \cdot 3 \cdot 5^2 \cdot 11 \cdot \ldots q_1 \cdot q_2 \cdot q_3 \cdot \ldots$, where $q_1 = 7623851$ (23 bits), $q_2 = 34110701$ (26 bits) and $q_3 = 2400573761$ (32 bits). Alice can use 2^e-isogenies for any $e \leq m$, and can subsequently scale her security up and down over the same field (e.g. to match the security of any of the SIKE instances). On Bob's side, he can compute L-isogenies for any $L \mid p-1$, e.g. with $L = \prod_{\ell_i \leq q_n} \ell_i^{e_i}$, he can take $n = 1$ to match SIKEp434, $n = 2$ to match SIKEp503, and $n = 3$ to match SIKEp610. Taking $m = 127$ is too small to offer any reasonable security in the elliptic curve setting, however combining the security analyses in [16, §4.1] and [12] reveals that B-SIDH construction in the genus-2 setting could achieve good post-quantum security over this smaller Mersenne prime. The factorisation $p - 1 = 2^{127} - 2 = 2 \cdot 3^3 \cdot 7^2 \cdot 19 \cdot 43 \cdot 73 \cdot 127 \cdot 337 \cdot 5419 \cdot 92737 \cdot 649657 \cdot 77158673929$ shows that the product of all odd primes up to 649657 (20 bits) could build a genus-2 isogeny that is large enough to obtain 128 bits of classical security and 64 bits of quantum security.

5.1.2. *The Ridinghoods.* Putting $c = 2^m - 1$ into (3) yields *Ridinghood* primes, which offer fast Karatsuba-style arithmetic in $\mathbb{F}_p$; the most famous of these has $c = 2^{224}$ and underlies Hamburg's Goldilocks curve [20]. Here Alice can meet the security offered by SIKEp434 by computing 2^{224}-isogenies. If Bob is to compute L-isogenies with $L \mid p - 1$, he would need to compute a prime isogeny whose degree is 78 bits in length. However, allowing Bob to work on both sides (by including factors of c) shows that he can meet the same requisite security when L's largest prime factor is only 24 bits. Of the other Ridinghoods with $m \in \{161, 208, 224, 225, 240, 354\}$, the most striking example is with $m = 225$; here the largest prime-degree isogeny needed for Bob to match the security of SIKEp434 is $\ell = 2^{16} + 1$. Note that both of these examples are subject to the caveat in §3.2.

5.1.3. *Montgomery-friendly primes.* While there are only a handful of Mersenne and Ridinghood primes, there are vastly more Montgomery-friendly primes, which are defined (see [5, §3.2] or [20]) as being precisely of the form in (3). For a fixed m and an upper bound on the bitlength of p, this gives many values of c that can be searched over until there is an $N \mid c(p - 1)$ that is both large enough for Bob to be secure, and smooth enough that he can compute the corresponding isogenies. A quick search with $m = 216$ (to match SIKEp434 exactly) and $1 \leq c < 2^{40}$ found two examples with $c = 137439113067$ and $c = 1030791469290$ where Bob's largest isogeny is of degree $\ell < 2^{30}$. To avoid the caveat in §3.2, similar searches could disallow factors of c so that Bob can work solely with torsion from the twists. Note that the search that found the above examples was far from exhaustive; there are likely to be much better examples with $m = 216$ and almost certainly better examples if m is varied. Alternatively, one can allow c to be larger (e.g. to occupy another machine word) in order to increase the search space and obtain better isogenies on Bob's side.

5.2. **Optimal public keys.** In terms of the size of the resulting public keys, the optimal scenario is when $M \approx N \approx p$. Recall from §3.1 that this boils down to searching for cryptographically sized primes p such that $p + 1$ and $p - 1$ are as smooth as possible. This subsection is focused on searching for such primes.

An earlier version of this paper aimed to find primes p such that $p-1$ and $p+1$ are minimally smooth by way of Störmer's theorem [37] (see also [25]). For a given smoothness bound B, Störmer's theorem says that are a finite number of integers, x, such that $x-1$ and $x+1$ are B-smooth; moreover, it gives a way to find this set in its entirety. If there are t primes up to B, then finding this set of integers amounts to solving all Pell equations of the form $x^2 - Dy^2 = 1$, where D is both squarefree and B-smooth; there are clearly 2^t such D, and therefore 2^t Pell equations to be solved [25]. Unfortunately, the sizes of B for which this task is feasible did not produce any values of x that offer meaningful security (at least, not in the case where the primes are chosen to underly *elliptic* curves). For example, with $B = 47$, the largest x such that $x-1$ and $x+1$ are B-smooth is (the 42-bit integer) $x = 2218993446251$. With $B = 113$, the largest such x is $x = 38632316754147847668001$ (76 bits), and the largest prime such x is $x = 410284126426376207279$ (69 bits); this required solving $2^t = 2^{30}$ Pell equations, and was the largest B exhaustively searched in this work.

Although it was infeasible to extend this method to the sizes of B required to produce $p > 2^{200}$, it did prove useful in showing factorisation patterns that often arose for values in the larger ranges. In particular, the largest prime values were often of the form $p = 2z^n - 1$, with z and n both integers, and where $n > 1$. Indeed, searching for primes of this form has proven to be the most useful method to date, and the reason is best illustrated via an example. With $n = 2$, we can search over B-smooth x such that $p = 2x^2 - 1$ is prime, at which point we are guaranteed that $p+1$ is B-smooth and we are hoping that $p-1 = 2x^2 - 2 = 2(x-1)(x+1)$ is also B-smooth. In other words, we are hoping that two values in $O(\sqrt{p})$ are B-smooth. In contrast, a naive search (i.e. a search with $n = 1$) would be hoping to find one value in $O(p)$ that is B-smooth. Under the heuristic assumption that $x-1$ and $x+1$ are uniformly distributed in $O(\sqrt{p})$, and taking into account well-established smoothness probabilities (cf. [26]), it becomes clear that the search with $n = 2$ is far superior.

This same reasoning extends to larger values of n, and it is readily seen that (for a fixed smoothness bound B and desired size of p) the success probability of the search becomes tied to the ratio d/n, where d is the degree of the largest irreducible factor(s) of $x^n - 1 \in \mathbb{Z}[x]$. Larger values of n can be chosen to minimise this ratio, however a larger n means fewer values of x to search over (for a desired size of $p = 2x^n - 1$). Though some examples were found with $n > 6$ (see §5.2.6), the *sweet spot* when aiming for primes between 192 and 256 bits proved to be $n = 4$ and $n = 6$.

The five examples presented in §5.2.1–§5.2.5 are summarised in Table 1. For a given prime p, it lists the bitlengths of the maximum isogeny degrees required by Alice and Bob, runtime complexities of the relevant classical and quantum attacks (written as base-2 logarithms), and the public key sizes of both standalone B-SIDH and a B-SIDH+ECDH hybrid. Following Section 4, the runtime of the Delfs-Galbraith (DG) algorithm is taken as $p^{1/2}$, the runtime of van Oorschot-Weiner (vOW) is taken as $2.5 \cdot p^{3/4}/2^{40}$, and the runtime of Biasse-Jao-Sankar (BJS) is taken as $p^{1/4}$; concrete runtimes in all three cases could be obtained by multiplying these complexities with the time taken for the corresponding isogeny computations. While the DG and BJS algorithms depend on the size of p, the complexity of the vOW algorithm depends on the number of possible isogenies computed by a given party (see §4.1). In the larger examples, Bob's use of all of the odd factors of $p-1$ can be overkill, so in these instances two options for Bob's isogenies and the subsequent vOW runtime estimates are given.

Following [11], B-SIDH public keys are three elements of $\mathbb{F}_{p^2}$, and partnering with an ECDH hybrid adds one additional element of $\mathbb{F}_p$ (the x-coordinate of the public key corresponding to a non-supersingular Montgomery curve with a strong ECDLP). It is worth pointing out that the asymptotic runtime of Delfs-Galbraith against B-SIDH matches the asymptotic runtime of Pollard rho [34] against the ECDLP, making the simplicity of the hybrid approach in [11, §8] particularly attractive.

TABLE 1. Summary of five B-SIDH-friendly primes p. Further explanation in text.

ex.	p (bits)	$\ell_{\text{Alice}}^{\max}$ (bits)	$\ell_{\text{Bob}}^{\max}$ (bits)	classical DG	classical vOW	quantum BJS	PK (bytes) B-SIDH	PK (bytes) hybrid
§5.2.1	195	10	13	98	108	49	147	171
§5.2.2	221	10	16	111	127	56	166	194
§5.2.3	237	6	20 17	119	139 125	60	178	208
§5.2.4	247	6	30 18	124	147 125	62	186	217
§5.2.5	250	12	19 16	125	149 122	63	188	219

5.2.1. *B-SIDHp195.* The 195-bit prime $p = 2 \cdot (2^2 \cdot 7 \cdot 127 \cdot 379 \cdot 431 \cdot 757 \cdot 887)^4 - 1$ has

$$\begin{aligned} p - 1 = 2 \cdot 3 \cdot 5 \cdot 13 \cdot 17 \cdot 19 \cdot 37 \cdot 53 \cdot 61 \cdot 173 \cdot 433 \cdot 449 \cdot 509 \cdot 613 \cdot 617 \cdot 677 \cdot 857 \\ \cdot 941 \cdot 1229 \cdot 2039 \cdot 2203 \cdot 2351 \cdot 2713 \cdot 3181 \cdot 7757. \end{aligned}$$

5.2.2. *B-SIDHp221.* The 221-bit prime $p = 2 \cdot (2^4 \cdot 13 \cdot 23 \cdot 127 \cdot 281 \cdot 607)^6 - 1$ has

$$\begin{aligned} p - 1 = 2 \cdot 3^2 \cdot 5 \cdot 7 \cdot 19 \cdot 31 \cdot 37 \cdot 43 \cdot 73 \cdot 163 \cdot 199 \cdot 211 \cdot 397 \cdot 571^2 \cdot 997 \\ \cdot 1307 \cdot 2161 \cdot 3181 \cdot 3449 \cdot 4861 \cdot 8821 \cdot 9319 \cdot 9511 \cdot 15061 \cdot 36691. \end{aligned}$$

5.2.3. *B-SIDHp237.* The 237-bit prime $p = 2 \cdot (2^3 \cdot 3^4 \cdot 17 \cdot 19 \cdot 31 \cdot 37 \cdot 53^2)^6 - 1$ has

$$\begin{aligned} p - 1 = 2 \cdot 7 \cdot 13 \cdot 43 \cdot 73 \cdot 103 \cdot 269 \cdot 439 \cdot 881 \cdot 883 \cdot 1321 \cdot 5479 \cdot 9181 \\ \cdot 12541 \cdot 15803 \cdot 20161 \cdot 24043 \cdot 34843 \cdot 48437 \cdot 62753 \cdot 72577 \cdot 709153. \end{aligned}$$

5.2.4. *B-SIDHp247.* The 247-bit prime $p = 2 \cdot (2^6 \cdot 3^2 \cdot 7^5 \cdot 11 \cdot 17 \cdot 31 \cdot 37)^6 - 1$ has

$$\begin{aligned} p - 1 = 2 \cdot 13 \cdot 19^2 \cdot 29 \cdot 43 \cdot 79 \cdot 83 \cdot 107 \cdot 643 \cdot 661 \cdot 733 \cdot 1447 \cdot 2347 \cdot 7753 \\ \cdot 28879 \cdot 29527 \cdot 38281 \cdot 64609 \cdot 76651 \cdot 86311 \cdot 228841 \cdot 745309897. \end{aligned}$$

5.2.5. *B-SIDHp250.* The 250-bit prime $p = 2 \cdot (5^3 \cdot 101 \cdot 211 \cdot 461 \cdot 2287)^6 - 1$ has

$$\begin{aligned} p - 1 = 2^4 \cdot 3^2 \cdot 7 \cdot 13 \cdot 37 \cdot 79 \cdot 107 \cdot 109 \cdot 199 \cdot 349 \cdot 433 \cdot 487 \cdot 1607 \cdot 1993 \cdot 3067 \\ \cdot 5701 \cdot 6199 \cdot 6373 \cdot 7883 \cdot 8821 \cdot 11497 \cdot 19507 \cdot 57037 \cdot 78301 \cdot 486839. \end{aligned}$$

5.2.6. *Larger n.* Although setting $n > 6$ shrinks the search space for primes $p = 2x^n - 1$ of a certain size, interesting examples were still found in some cases. These typically have p much larger than the degree of feasible isogenies on Bob's side, so fall back into the umbrella of the types of primes explored in §5.1 (here there is typically a comfortable enough margin between p and the isogeny degrees that claw-finding goes back to being the best classical attack). For brevity, write ℓ as the largest prime factor of a given $N \mid p - 1$ in each case. The 331-bit prime $p = 2 \cdot (3^2 \cdot 13)^{48} - 1$ has $N > 2^{213}$ with $\ell < 2^{23}$. The 367-bit prime $p = 2 \cdot (3^2 \cdot 127)^{36} - 1$ has $N > 2^{216}$ with $\ell < 2^{22}$. The 354-bit prime $p = 2 \cdot (2 \cdot 5 \cdot 7^3)^{30} - 1$ has $N > 2^{201}$ with $\ell < 2^{23}$. The 362-bit prime $p = 2 \cdot (2 \cdot 11^2 \cdot 17)^{30} - 1$ has $N > 2^{208}$ and the 363-bit $p = 2 \cdot (2^3 \cdot 23^2)^{30} - 1$ with $N > 2^{212}$, both with $\ell < 2^{24}$. The 258-bit prime $p = 2 \cdot (2^3 \cdot 3^2 \cdot 23)^{24} - 1$ has $N > 2^{229}$ with $\ell < 2^{21}$. The 325-bit prime $p = 2 \cdot (2 \cdot 3 \cdot 5 \cdot 13 \cdot 29)^{24} - 1$ has $N > 2^{270}$ with $\ell < 2^{26}$ and $N > 2^{220}$ with $\ell < 2^{21}$. The 250-bit prime $p = 2 \cdot (29 \cdot 31 \cdot 1901)^{12} - 1$ has $N > 2^{211}$ with $\ell < 2^{18}$ and N's largest factor is 20 bits, which is slightly worse for Bob than B-SIDHp250.

References

[1] G. Adj, D. Cervantes-Vázquez, J. Chi-Domínguez, A. Menezes, and F. Rodríguez-Henríquez. On the cost of computing isogenies between supersingular elliptic curves. In *SAC 2018*, pages 322–343. Springer, 2018.

[2] R. Azarderakhsh, D. Jao, K. Kalach, B. Koziel, and C. Leonardi. Key compression for isogeny-based cryptosystems. In *AsiaPKC*, pages 1–10. Springer, 2016.

[3] D. J. Bernstein. Curve25519: new Diffie-Hellman speed records. In *International Workshop on Public Key Cryptography*, pages 207–228. Springer, 2006.

[4] J. Biasse, D. Jao, and A. Sankar. A quantum algorithm for computing isogenies between supersingular elliptic curves. In *INDOCRYPT 2014*, pages 428–442. Springer, 2014.

[5] J. W. Bos and P. L. Montgomery. *Montgomery Arithmetic from a Software Perspective*, page 10–39. Cambridge University Press, 2017.

[6] W. Castryck, T. Lange, C. Martindale, L. Panny, and J. Renes. CSIDH: an efficient post-quantum commutative group action. In *ASIACRYPT 2018*, pages 395–427. Springer, 2018.

[7] D. Cervantes-Vázquez, E. Ochoa-Jiménez, and F. Rodríguez-Henríquez. eSIDH: the revenge of the SIDH. Preprint, 2020. `https://eprint.iacr.org/2020/021`.

[8] A. M. Childs, D. Jao, and V. Soukharev. Constructing elliptic curve isogenies in quantum subexponential time. *J. Mathematical Cryptology*, 8(1):1–29, 2014.

[9] C. Costello and H. Hisil. A simple and compact algorithm for SIDH with arbitrary degree isogenies. In *ASIACRYPT 2017*, pages 303–329. Springer, 2017.

[10] C. Costello, D. Jao, P. Longa, M. Naehrig, J. Renes, and D. Urbanik. Efficient compression of SIDH public keys. In *EUROCRYPT 2017*, pages 679–706. Springer, 2017.

[11] C. Costello, P. Longa, and M. Naehrig. Efficient algorithms for supersingular isogeny Diffie-Hellman. In *CRYPTO 2016*, pages 572–601. Springer, 2016.

[12] C. Costello and B. Smith. The supersingular isogeny problem in genus 2 and beyond. *PQCrypto2020, to appear. https://eprint.iacr.org/2019/1387*.

[13] J. M. Couveignes. Hard homogeneous spaces. Preprint, 2006. `http://eprint.iacr.org/2006/291`.

[14] C. Delfs and S. D. Galbraith. Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$. *Designs, Codes and Cryptography*, 78(2):425–440, 2016.

[15] L. De Feo, D. Jao, and J. Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *J. Mathematical Cryptology*, 8(3):209–247, 2014.

[16] E. V. Flynn and Y. B. Ti. Genus two isogeny cryptography. In *PQCrypto 2019*, pages 286–306. Springer, 2019.

[17] S. D. Galbraith, C. Petit, B. Shani, and Y. B. Ti. On the security of supersingular isogeny cryptosystems. In *ASIACRYPT 2016*, pages 63–91. Springer, 2016.

[18] S. D. Galbraith, C. Petit, and J. Silva. Identification protocols and signature schemes based on supersingular isogeny problems. In *ASIACRYPT 2017*, pages 3–33. Springer, 2017.

[19] L. K. Grover. A fast quantum mechanical algorithm for database search. In *STOC 1996*, pages 212–219. ACM, 1996.

[20] M. Hamburg. Ed448-Goldilocks, a new elliptic curve. *IACR Cryptology ePrint Archive*, 2015:625, 2015.

[21] 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. SIKE: Supersingular Isogeny Key Encapsulation. Manuscript available at `sike.org/`, 2017.

[22] D. Jao and L. De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In *PQCrypto 2011*, pages 19–34. Springer, 2011.

[23] S. Jaques and J. M. Schanck. Quantum cryptanalysis in the RAM model: Claw-finding attacks on SIKE. In *CRYPTO 2019*, pages 32–61. Springer, 2019.

[24] G. Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM J. Comput.*, 35(1):170–188, 2005.

[25] D. H. Lehmer. On a problem of Störmer. *Illinois Journal of Mathematics*, 8(1):57–79, 1964.

[26] Arjen K. Lenstra. Smoothness probability. In *Encyclopedia of Cryptography and Security*. 2005.

[27] K. Matsuo. SIDH over quadratic twists. In *Proc. of SCIS2019 -2019 Symposium on Cryptography and Information Security, 3B3-1*, January 2019 `https://www.iwsec.org/scis/2019/`.

[28] M. Meyer and S. Reith. A faster way to the CSIDH. In *INDOCRYPT 2018*, pages 137–152. Springer, 2018.

[29] P. L. Montgomery. Speeding the Pollard and elliptic curve methods of factorization. *Mathematics of computation*, 48(177):243–264, 1987.

[30] M. Naehrig and J. Renes. Dual isogenies and their application to public-key compression for isogeny-based cryptography. In *ASIACRYPT 2019*, pages 243–272. Springer, 2019.

[31] C. Petit. Faster algorithms for isogeny problems using torsion point images. In *ASIACRYPT 2017*, pages 330–353. Springer, 2017.

[32] A. K. Pizer. Ramanujan graphs and Hecke operators. *Bulletin of the American Mathematical Society*, 23(1):127–137, 1990.

[33] A. K. Pizer. Ramanujan graphs. *AMS/IP Stud. Adv. Math.*, 7:159–178, 1998.

[34] J. M. Pollard. Monte Carlo methods for index computation (mod p). *Mathematics of computation*, 32(143):918–924, 1978.

[35] A. Rostovtsev and A. Stolbunov. Public-key cryptosystem based on isogenies. Preprint, 2006. `https://eprint.iacr.org/2006/145`.

[36] J. H. Silverman. *The Arithmetic of Elliptic Curves, 2nd Edition.* Graduate Texts in Mathematics. Springer, 2009.

[37] C. Størmer. Quelques théorèmes sur l'équation de Pell $x^2 - dy^2 = \pm 1$ et leurs applications. *Christiania Videnskabens Selskabs Skrifter, Math. Nat. Kl*, (2):48, 1897.

[38] S. Tani. Claw finding algorithms using quantum walk. *Theor. Comput. Sci.*, 410(50):5285–5297, 2009.

[39] J. Tate. Endomorphisms of abelian varieties over finite fields. *Inventiones mathematicae*, 2(2):134–144, 1966.

[40] P. C. van Oorschot and M. J. Wiener. Parallel collision search with cryptanalytic applications. *J. Cryptology*, 12(1):1–28, 1999.

[41] J. Vélu. Isogénies entre courbes elliptiques. *CR Acad. Sci. Paris Sér. AB*, 273:A238–A241, 1971.

[42] C. Zalka. Grover's quantum searching algorithm is optimal. *Physical Review A*, 60(4):2746, 1999.

[43] G. Zanon, M. A. Simplício Jr., G. C. C. F. Pereira, J. Doliskani, and P. S. L. M. Barreto. Faster key compression for isogeny-based cryptosystems. *IEEE Trans. Computers*, 68(5):688–701, 2019.