

New SIDH Countermeasures for a More Efficient Key Exchange

Andrea Basso¹[0000–0002–3270–1069] and Tako Boris Fouotsa²[0000–0003–1821–8406]

¹ University of Bristol, Bristol, United Kingdom

`andrea.basso@bristol.ac.uk`

² EPFL, Lausanne, Switzerland

`tako.fouotsa@epfl.ch`

Abstract. The Supersingular Isogeny Diffie-Hellman (SIDH) protocol has been the main and most efficient isogeny-based encryption protocol, until a series of breakthroughs led to a polynomial-time key-recovery attack. While some countermeasures have been proposed, the resulting schemes are significantly slower and larger than the original SIDH.

In this work, we propose a new countermeasure technique that leads to significantly more efficient and compact protocols. To do so, we introduce the concept of artificially oriented curves, which are curves with an associated pair of subgroups. We show that this information is sufficient to build parallel isogenies and thus obtain an SIDH-like key exchange, while also revealing significantly less information compared to previous constructions.

After introducing artificially oriented curves, we formalize several related computational problems and thoroughly assess their presumed hardness. We then translate the SIDH key exchange to the artificially oriented setting, obtaining the key-exchange protocols `binSIDH`, or binary SIDH, and `terSIDH`, or ternary SIDH, which respectively rely on fixed-degree and variable-degree isogenies.

Lastly, we also provide a proof-of-concept implementation of the proposed protocols. Despite being implemented in a high-level language, `terSIDH` has very competitive running times, which suggests that `terSIDH` might be the most efficient isogeny-based encryption protocol.

1 Introduction

Given two elliptic curves, finding an isogeny between them is widely believed to be a computationally hard problem. This has led to the development of several cryptographic protocols, whose security relies on the hardness of some isogeny-related problem. While the first constructions date back to 1996 [22], the first practical isogeny-based protocol was the Supersingular Isogeny Diffie-Hellman (SIDH) key exchange [35]. After a decade of improvements and analysis, the protocol became the most efficient and well-known encryption scheme from isogenies, and it progressed through the four rounds of the NIST standardization process.

The security of the protocol, however, did not rely on the pure isogeny problem: finding an isogeny between two supersingular elliptic curves. The problem is hard, but its lack of structure makes it hard to obtain cryptographic functionalities off it. Thus, SIDH needed to reveal additional information in the form of torsion images: not only were the domain and codomain of the secret isogenies known, but also their actions on a torsion subgroup of coprime order. This additional information has been studied over the years, and it has been shown to lead to some active attacks [33] and key-recovery attacks when the endomorphism ring of the two curves is known [33] or when the protocol uses unbalanced parameters [40,42]. However, all these attacks came short of affecting the security of SIDH. The situation changed when a series of works [9,38,43] developed a polynomial-time attack against SIDH for all possible parameters.

These attacks do not affect the security of other isogeny-based protocols, such as CSIDH [11] and SQISign [27], but they affect those protocols that reveal images of torsion points, such as SÉTA [24]. Some countermeasures against the SIDH attacks have been proposed [29]: they are based on scaling the torsion images (M-SIDH) or computing variable-degree isogenies (MD-SIDH). However, the complexity of the attacks against these protocols scale with the number of distinct primes dividing the isogeny degrees: thus, to be secure, these protocols require extremely large parameters, which lead to high running times and communication costs.

Besides M-SIDH and MD-SIDH, the only currently secure encryption protocols based on isogenies are CSIDH [11] and FESTA [5]. However, the first is vulnerable to a subexponential quantum attack [39], which makes it hard to estimate the quantum security of a given parameter set. The more conservative estimates require large primes, which lead to impractically inefficient running times [16]. The second, FESTA, is a recent public-key encryption protocol based on a constructive application of the SIDH attacks. While the initial results are promising, the protocol computes isogenies between elliptic curves of large prime degree (around 2^{16}) and isogenies between abelian varieties, which both affect the efficiency of the protocol. A third key-exchange protocol, pSIDH [37], offered interesting properties but was recently broken by Chen, Imran, Ivanyos, Kutas, Leroux and Petit [14].

In this work, we aim to fill the gap by proposing new countermeasures against the SIDH attacks that lead to a practically efficient SIDH-like key-exchange protocol. To do so, we introduce the concept of artificial orientations: an artificial A -orientation \mathfrak{A} on a supersingular elliptic curve E is a pair of cyclic disjoint subgroups of $E[A]$ of order A . Given an artificial orientation $\mathfrak{A} = (G_1, G_2)$, an \mathfrak{A} -isogeny ϕ is an isogeny whose kernel is the direct sum of a cyclic subgroup of G_1 and a cyclic subgroup of G_2 . In other words, ϕ can be written as the composition $\phi = \phi_2 \circ \phi_1$, where $\ker \phi_1 \subset G_1$, $\ker \phi_2 \subset \phi_1(G_2)$, and the degrees of ϕ_1 and ϕ_2 are coprime. While an artificial orientation does not reveal the same information as a standard orientation [18], it provides an interpolation between the original SIDH construction and the oriented protocols, such as CSIDH [11], OSIDH [18], and SCALLOP [26]. On one hand, artificial orientations and their images provide

enough information to compute parallel isogenies, similarly to torsion images in SIDH; on the other, orientations always imply an artificial orientation, because given an orientation it is possible to recover the images of two cyclic disjoint groups, i.e. an artificial orientation. For example, in CSIDH, the images of the groups $\ker(\pi - 1) \cap E[\ell]$ and $\ker(\pi + 1) \cap E[\ell]$ under the secret isogeny $\phi : E \rightarrow E'$ are given by $\ker(\pi - 1) \cap E'[\ell]$ and $\ker(\pi + 1) \cap E'[\ell]$, respectively [10, Section 6.1].

Contributions. In this paper, we formalize the concept of artificial orientations and introduce some computational problems related to artificially oriented isogenies. We thoroughly assess the presumed hardness of these problems and we survey potential attacks. Then, we propose `binSIDH`, or binary SIDH, the first protocol that translates SIDH to the artificially-oriented setting. In other words, both parties compute an oriented isogeny, one whose kernel is the direct sum of two cyclic subgroups of the orientation, and reveal the images of a second orientation under the secret isogeny. This allows both parties to obtain a shared secret while revealing significantly less information.

As in SIDH, such a key exchange is limited to fixed-degree isogenies, which is helpful to develop constant-time implementations and zero-knowledge proofs of isogeny knowledge. Then, we generalize `binSIDH` to the case of variable-degree isogenies to obtain `terSIDH`, or ternary SIDH, which achieves smaller parameters.

The two protocols, `binSIDH` and `terSIDH`, require both parties to use artificially oriented isogenies, which results in a balanced protocol where the computational requirements of both parties is similar. We also propose a new technique that allows one party to compute SIDH-like isogenies, at the cost of the other party computing longer oriented ones. This allows one party to be significantly more efficient, which is particularly useful in advanced protocols between clients and servers with unbalanced computational power: not only can the client be more efficient than the server, but if the protocol requires proofs of isogeny knowledge, those of the client can be computed much more efficiently as well. Since the same technique can be applied to `binSIDH` and `terSIDH`, we obtain two new variants: `binSIDHhyb` and `terSIDHhyb`.

Lastly, we generate parameter sets for all four protocols, for all security levels, and we provide a SageMath proof-of-concept implementation of all proposed protocols. Despite being implemented in a high-level language, `terSIDH` has very competitive running times when compared to existing implementations of other isogeny-based encryption schemes.

2 Preliminaries

In this section, we briefly introduce the SIDH protocol and the recent key-recovery attacks. For more background information on elliptic curves and isogenies, we refer the reader to [44].

2.1 SIDH

SIDH, or Supersingular Isogeny Diffie-Hellman [35], is a key-exchange protocol based on isogenies between supersingular elliptic curves. The main protocol parameters are a prime p of the form $p = ABf - 1$, where $A = 2^a$ and $B = 3^b$, and a starting supersingular elliptic curve E_0 defined over \mathbb{F}_{p^2} . The protocol also specifies two bases P_A, Q_A and P_B, Q_B that generate, respectively, $E_0[A]$ and $E_0[B]$.

The first party, say Alice, generates her public key by sampling a random secret key $\text{sk}_A = \alpha \in \mathbb{Z}_A$, computing the isogeny $\phi_A : E_0 \rightarrow E_A$ with kernel $\ker \phi_A = \langle P_A + [\alpha]Q_A \rangle$, and revealing $\text{pk}_A = (E_A, R_A = \phi_A(P_B), S_A = \phi_A(Q_B))$. The second party, say Bob, proceeds analogously with an isogeny of degree B : he samples $\text{sk}_B = \beta \in \mathbb{Z}_B$, computes the isogeny $\phi_B : E_0 \rightarrow E_B$ with kernel $\ker \phi_B = \langle P_B + [\beta]Q_B \rangle$, and reveals $\text{pk}_B = (E_B, R_B = \phi_B(P_A), S_B = \phi_B(Q_A))$. Then, after exchanging public keys, both parties can obtain the same shared secret by computing the push-forward of their isogeny under the other party's isogeny. Concretely, Alice computes the isogeny $\phi'_A : E_B \rightarrow E_{AB}$ with kernel $\ker \phi'_A = \langle R_B + [\alpha]S_B \rangle = \phi_B(\ker \phi_A)$, while Bob computes the isogeny $\phi'_B : E_A \rightarrow E_{BA}$ with kernel $\ker \phi'_B = \langle R_A + [\beta]S_A \rangle = \phi_A(\ker \phi_B)$. The two isogenies are the correct push-forwards, and thus $\phi_A, \phi_B, \phi'_A, \phi'_B$ form a commutative diagram. Hence, the codomain curves E_{AB} and E_{BA} are isomorphic, and their j -invariant is the shared secret known to both Alice and Bob.

2.2 Polynomial time attacks on SIDH

The security of the SIDH protocol relies on the hardness of recovering a secret isogeny from its action on a torsion basis. In a series of works by Castryck and Decru [9], Maino, Martindale, Panny, Pope and Wesolowski [38], and Robert [43], the authors show the problem can be solved in polynomial time when the torsion information is sufficiently large compared to the degree of the isogeny. This leads to an efficient key-recovery attack on all instances of SIDH.

The attacks slightly vary in their techniques, but they all rely on Kani's theorem [36], which implies that given an SIDH square with specific properties, there exists an isogeny between the principally polarized abelian surface obtained by gluing two curves in the SIDH square to the principally polarized abelian surface obtained by gluing the other two curves in the square. It is possible to generate an SIDH square with the desired properties and compute the genus-two isogeny from the image points revealed in SIDH; evaluating such an isogeny allows an attacker to evaluate the secret isogeny on any point, which in turn can be used to recover the secret isogeny.

For the purpose of this work, the SIDH attacks can be abstracted as a generic algorithm that recovers an isogeny $\phi : E_0 \rightarrow E_1$ of degree d when it receives the curves E_0, E_1 , the degree d , and the points P_0, Q_0 and $\phi(P_0), \phi(Q_0)$, where P_0, Q_0 are linearly independent points of order n and $n^2 > 4d$. There is no known technique that allows extending such attacks to a case where the image points are not known exactly: indeed, all attacks on the proposed countermeasures [29],

as well as the potential attacks discussed in this work, need to recover the exact torsion images to apply the attacks.

3 Artificial orientations

In this section, we introduce artificial orientations, the main ingredient that powers the countermeasures against the SIDH attacks. In the rest of the paper, the integers A and B are assumed to be smooth, coprime with each other, and square-free.³ They also define a prime p of the form $p = ABf - 1$, where f is a small cofactor needed for primality; thus, the values A and B are always smaller than p and coprime with it.

Artificial orientations are composed of two independent subgroups. This is formalized in [Definition 1](#), and we provide more information on how to explicitly compute such isogenies in [Eq. \(2\)](#) in [Section 4.1](#).

Definition 1. *Let E be a supersingular elliptic curve defined over \mathbb{F}_{p^2} , and let A be an integer. An artificial A -orientation (of E) is a pair $\mathfrak{A} = (G_1, G_2)$ where $G_1, G_2 \subset E[A]$ are cyclic groups of order A and $G_1 \cap G_2 = \{0\}$. (E, \mathfrak{A}) is called an artificially A -oriented curve.*

Remark 1. Artificial orientations are known in the number theory literature as *split Cartan level structures*. We prefer the artificial orientation name as it may be more familiar to those who know isogeny-based cryptography, but the connection may lead to useful insights. For instance, see [\[17\]](#) for an analysis of the mixing properties of the isogeny graph with split Cartan level structures.

Given an artificially A -oriented curve (E, \mathfrak{A}) , one can compute a range of isogenies whose kernels arise from $\mathfrak{A} = (G_1, G_2)$. We formalize this concept, which we call \mathfrak{A} -isogenies, in the following definition.

Definition 2. *Let (E, \mathfrak{A}) where $\mathfrak{A} = (G_1, G_2)$ be an artificially A -oriented curve. An isogeny $\phi : E \rightarrow E'$ is said to be an \mathfrak{A} -isogeny if $\ker \phi$ is the direct sum of a subgroup of G_1 and a subgroup of G_2 , that is $\ker \phi = H_1 \oplus H_2$ where H_i is a subgroup of G_i for $i = 1, 2$.*

If (E, \mathfrak{A}) is an artificially A -oriented curve and $\phi : E \rightarrow E'$ is a non-trivial \mathfrak{A} -isogeny, then the artificial A -orientation on E cannot be carried onto E' through ϕ . In fact, since ϕ is non-trivial and $\ker \phi$ is the direct sum of a subgroup of G_1 and a subgroup of G_2 , then at least one of the groups $\phi(G_1)$ and $\phi(G_2)$ has order strictly smaller than A . In order to be able to carry the artificial A -orientation on E onto E' it is necessary that the degree of the isogeny considered is coprime to A . We have the following definition for artificially A -oriented B -isogenous curves.

³ The square-free property is not necessary for the correctness of the protocols, but square divisors of A and B decrease the efficiency of the protocols without increasing their security

Definition 3. Let (E, \mathfrak{A}) and (E', \mathfrak{A}') be two artificially A -oriented curves and let B be an integer coprime to A . We say that (E, \mathfrak{A}) and (E', \mathfrak{A}') are B -isogenous if there exists an isogeny $\phi : E \rightarrow E'$ of degree B such that $\mathfrak{A}' = \phi(\mathfrak{A})$, that is if $\mathfrak{A} = (G_1, G_2)$ and $\mathfrak{A}' = (G'_1, G'_2)$, then $G'_1 = \phi(G_1)$ and $G'_2 = \phi(G_2)$.

Remark 2. Note that B -isogenous oriented curves include images of subgroups. These can be represented by choosing a random generator. Thus, if we fix generators $\langle P_1 \rangle = G_1$ and $\langle P_2 \rangle = G_2$, the subgroups G'_1 and G'_2 are represented by $[\alpha]\phi(P_1)$ and $[\beta]\phi(P_2)$ respectively, for some unknown $\alpha, \beta \in \mathbb{Z}_A$.

3.1 A comparison of \mathfrak{A} -isogenies with existing techniques

In this section, we discuss the main differences between artificially oriented isogenies and fully oriented isogenies, such as those used in CSIDH and SCALLOP, and between oriented isogenies and the more generic isogenies used in SIDH, M-SIDH, and MD-SIDH.

\mathfrak{A} -isogenies vs group actions. Artificially oriented isogenies share similarities with those that arise from group actions, such as the isogenies in CSIDH, OSIDH, and SCALLOP. In both instances, isogenies are restricted to specific subsets of all possible isogenies, and the action of secret isogenies on two independent subgroups is revealed [10, Section 6.1]. However, artificially oriented isogenies are significantly different from those in CSIDH and SCALLOP: first, given any supersingular elliptic curve, it is always possible to attach an artificial orientation to it, unlike in CSIDH, where the curves need to be defined over \mathbb{F}_p and the orientation is already available through the Frobenius endomorphism; or SCALLOP, where not all supersingular elliptic curves are oriented and a standard orientation needs to be provided. Most importantly, artificial orientations do not give rise to a commutative group action as the one from standard orientations, which means that the quantum subexponential attack by Childs, Jao, and Soukharev [15] does not apply. Similarly, artificial orientations are also immune to the attacks on OSIDH [23].

\mathfrak{A} -isogenies vs SIDH. The main difference between SIDH-like isogenies and artificially A -oriented isogenies is the amount of information needed to compute their push-forwards. In the SIDH case, the kernel of the isogeny (say ψ) is generated by a point of the form $P + [\alpha]Q$. The kernel of the push-forward of ψ through ϕ is generated by the point $\phi(P) + [\alpha]\phi(Q)$. Therefore, the images of torsion points P and Q are needed in order to compute the push-forward of ψ through ϕ . Conversely, \mathfrak{A} -isogenies are limited to those that arise from \mathfrak{A} . Hence, only the push-forward of the artificial orientation is needed, which means only the images of two cyclic torsion groups are revealed. This prevents torsion point attacks [40,42,9,38,43].

\mathfrak{A} -isogenies vs M-SIDH. In M-SIDH and MD-SIDH [29], isogenies are defined as in SIDH, but to compute their push-forwards, the torsion points images are revealed while scaled (or masked) with the same scalar β . This means that instead of revealing $\phi(P)$ and $\phi(Q)$ as in SIDH, one reveals $[\beta]\phi(P)$ and $[\beta]\phi(Q)$. This is significantly more information than what is revealed to compute the push-forwards of \mathfrak{A} -isogenies, since the image of an artificial orientation is equivalent, as discussed in [Remark 2](#), to revealing $[\alpha]\phi(P)$ and $[\beta]\phi(Q)$, for independent values α and β . From a subgroup perspective, push-forwards of \mathfrak{A} -isogenies require the images of two cyclic disjoint subgroups, whereas M-SIDH reveals two image points scaled with the same value, which is equivalent to the images of three cyclic disjoint groups of order $\text{ord}(P)$ (see [4, Lemma 1] and [30, Lemma 1]).

3.2 Security assumptions

Having introduced artificial orientations, we now introduce three computational problems that relate to artificially oriented curves and isogenies. The first problem, which we refer to as the Supersingular Isogeny Problem for artificially A -oriented curves (SSIP- A), asks to recover an isogeny given its domain, together with an artificial orientation, and its codomain, together with a compatible orientation. This corresponds to finding a path of length B in the isogeny graph with split Cartan level structure.

Problem 1 (SSIP- A). Let (E, \mathfrak{A}) be an artificially A -oriented curve and let B be an integer coprime to A . Let $\phi : E \rightarrow E'$ be a cyclic isogeny of degree B and let $\mathfrak{A}' = \phi(\mathfrak{A})$. Given (E, \mathfrak{A}) and (E', \mathfrak{A}') and the degree B , compute ϕ .

In [Problem 1](#), there is no constraint on the isogeny ϕ , apart from its degree being B . When an artificial B -orientation \mathfrak{B} is provided on E , then one may restrict to \mathfrak{B} -isogenies. This leads to the (supersingular) Artificially Oriented Isogeny Problem (AOIP).

Problem 2 (AOIP). Let (E, \mathfrak{A}) an artificially A -oriented curve and let B be an integer coprime to A . Let \mathfrak{B} be an artificial B -orientation on E . Let $\phi : E \rightarrow E'$ be a cyclic \mathfrak{B} -isogeny of degree B and let $\mathfrak{A}' = \phi(\mathfrak{A})$. Given $(E, \mathfrak{A}, \mathfrak{B})$ and (E', \mathfrak{A}') , compute ϕ .

We can also study a problem that is, in some sense, the converse of [Problem 1](#). Rather than considering general isogenies and the image of an artificial orientation, we can focus on the case where the isogeny is artificially oriented, but more torsion image information is revealed. This is summarized in the Supersingular Isogeny Problem for \mathfrak{B} -isogenies (SSIP- B) problem.

Problem 3 (SSIP- B). Let (E, \mathfrak{A}) be an artificially A -oriented curve and let B be an integer coprime to A . Let $\phi : E \rightarrow E'$ be a cyclic \mathfrak{A} -isogeny of degree A , with $B \ll A$. Let also P, Q be a basis of $E[B]$. Given (E, \mathfrak{A}) , together with the points P, Q , and the curve E' with the points $\phi(P)$ and $\phi(Q)$, compute ϕ .

If $B \not\ll A$, such a problem could be solved with the techniques introduced in the SIDH attacks [9,38,43]. However, for larger choices of B (when compared to A), **Problem 3** is believed to be secure, and its hardness can be used to build more efficient protocols, as we will see in **Section 5**.

3.3 Hardness analysis

In this section, we study the computational problems that we introduced, analyze potential attacks, and justify their assumed hardness.

Finding an isogeny from the orientation image. The first problem, **Problem 1** is already known in the literature, as it was recently introduced with a different notation in [5, Problem 7], where it was called the Computational isogeny with scaled-torsion (CIST) problem. As argued in [5], the problem appears to be hard because the images of two subgroups do not provide enough information for the SIDH attacks to be applicable. Given two images $[\alpha]\phi(P)$ and $[\beta]\phi(Q)$, scaled by independent values α and β , an attacker can easily recover the product $\alpha\beta$ from pairing computations, but this is similarly insufficient to recover the exact images that would enable the SIDH attacks. An attacker may attempt to brute force the missing information, but this is computationally infeasible if the degree of the secret isogeny is sufficiently large, which in turn makes the order of the torsion information to be guessed large enough for the attack to be infeasible. Note that the information revealed in **Problem 1** is comparable to that in CSIDH and SCALLOP, and significantly less than that in M-SIDH and MD-SIDH. It is thus likely that an attack that can solve **Problem 1** in its most general form, can do so for such protocols as well.

Since not enough information is revealed for the SIDH attacks to apply, the attack on starting curves with small endomorphisms [29] does also not apply here. It is thus possible to choose a starting curve with known endomorphism ring. Very recent analysis [12] has shown it is possible to recover an isogeny from its scaled action and thus solve **Problem 1** when the starting curve E_0 and the corresponding orientation has specific properties relative to the Frobenius conjugate $E_0^{(p)}$ of E_0 . It is thus important to select parameters that avoid these issues; since the endomorphism ring of the starting curve can be public, this can be done in a transparent manner without the need of a trusted setup. We further expand on the choice of the artificial orientation and of the starting curve at the end of this section.

Finding an oriented isogeny from the orientation image. In **Problem 2**, the degree of the isogeny ϕ is not necessarily known: the degree of a \mathfrak{B} -isogeny can range across all values dividing the order B of the subgroups in \mathfrak{B} , which poses a first barrier to the application of the SIDH attacks. However, even if we restrict to isogenies of full degree, i.e. $\deg \phi = B$, the torsion information that is revealed is the same as that in **Problem 1**, and thus a similar analysis follows. The fact that the unknown isogeny is a \mathfrak{B} -isogeny does not interact in

any meaningful way with the SIDH attacks or the revealed torsion information: as such, it appears to be hard for an attacker to exploit such attacks to solve [Problem 2](#). Hence, it seems likely that any attack would have to disregard the artificial orientation and focus on recovering an isogeny between two given curves; however, since the isogeny is a \mathfrak{B} -isogeny, this problem is easier than the general case.

First, an attacker can simply brute force all the possible isogenies. If we restrict ourselves to isogenies of full degrees, there are 2^t possible \mathfrak{B} -isogenies, where t is the numbers of primes dividing B . This suggests that the degree of the isogeny should be the product of at least $t = \lambda$ distinct primes. Second, generic attacks to recover an isogeny between two given curves, such as the meet-in-the-middle (MITM), van Oorschot-Wiener (vOW) [46], Delfs-Galbraith [28] attacks, are not applicable since the prime characteristic and the isogeny degree, being the product of at least λ distinct primes, are sufficiently large to make these attacks computationally infeasible. However, it is possible to devise an enhanced MITM attack that exploits the nature of the \mathfrak{B} -isogenies: the attacker fixes an attack parameter $0 \leq t' \leq t$ and then computes $2^{t'}$ \mathfrak{B} -isogenies starting from E_0 . These are chosen of the largest degree, i.e. the attacker first computes the isogenies with degree corresponding to the largest primes dividing B , so that the end curves are as close to E' as possible. The attacker stores the j -invariants of the codomain curves and starts a random walk of the correct degree from E' , in the hope of finding a collision. The cost of the attack depends on the choice of t' : the first part requires $2^{t'}$ computations, while the second part requires computing all the possible isogenies of a specific degree (the product of the smaller $t - t'$ primes dividing B , assuming that B is square-free) starting from E' . This technique yields a better attack than a simple brute-force approach, and thus it would require larger parameters, albeit only moderately larger ones.

Example 1. For instance, when B is the product of the first 128 primes (the case most suitable to this attack), the attack is optimal for $t = 106$, since such a value minimizes the product of the costs of the MITM attack and the brute-force attack. Setting $t = 106$ corresponds to an attack where 2^{106} isogenies are computed and 2^{106} j -invariants are stored in memory. Thus, to obtain $\lambda = 128$ bits of security, we would need the \mathfrak{B} -isogeny to have a degree B that is the product of the $t = 154$ smallest primes. This ensures that the optimal attack requires $t > 128$. We remark that the security estimates depend not only on the number of distinct primes dividing B , but also on the size of the specific primes.

The previous attack considers an attacker that has accesses to unbounded memory. This is far from realistic, and we can obtain better estimates of the attack possibilities when we impose an upper bound to the amount of memory available. We follow the security analysis of SIDH [1,34], and we limit our analysis to attackers with 2^{80} units of memory for any security level.⁴ In this setting, the best attack is a vOW version of the enhanced MITM attack presented

⁴ More precisely, we consider attackers that can store up to 2^{80} j -invariants. Given the size of the primes used, this corresponds to more than 2^{90} bits of memory.

before, which allows the attacker to trade higher computational costs for a lower memory requirement. As shown in [1], a vOW search has a computational cost of approximately

$$N^{3/2}/w^{1/2},$$

where N is the number of collision points and w is the number of memory units available. In our case, we have $N = 2^{t'}$, and $w = 2^{80}$. This suggests that, for $\lambda = 128$, this attack outperforms a brute-force search, but only marginally. If we set the degree B to be the product of the first t distinct primes, the enhanced vOW attack requires $t = 137$ (compared to $t = 128$, as suggested by the brute-force attack). However, for higher security levels, the brute-force attack outperforms the enhanced vOW attack, because the memory bound remains constant across all security levels, and thus it has a larger performance impact on higher security levels. Thus, for $\lambda \in \{192, 256\}$, we can choose $t = \lambda$.

The case with variable-degree isogenies follows similarly. A brute-force approach requires an attacker to compute isogenies starting from all previously visited curves, rather than just the end ones. In other words, the attack starts with a set of visited curves $\mathcal{S} = \{E_0\}$; for each prime p_i , the attacker computes two isogenies starting for all curves in \mathcal{S} and adds the codomain curves to \mathcal{S} . We restrict ourselves to the case where the maximum degree is square free, as we will choose for the parameters of terSIDH, since higher powers decrease efficiency without improving security. In this case, the complexity of the attack is the same as a brute-force attack with fixed-degree isogenies with three options at each node.

Overall, the rest of the attack proceeds similarly: as in the binSIDH case, the parameter t needs to be selected to avoid a brute-force attack, where the specific value depends on the exponents of the primes dividing B . The enhanced MITM and vOW attacks similarly apply to the variable-degree case: in this case, however, the enhanced vOW attack outperforms a brute-force attack at all commonly used security levels, and thus the parameters need to be slightly larger than what a brute-force attack would suggest.

Finding an isogeny from the full torsion image. Lastly, [Problem 3](#) is vulnerable to the SIDH attacks, as discussed when introduced. However, the \mathcal{A} -isogeny needs to have a large degree A to be secure from the attacks outlined above, and thus the torsion points would need a large order B for the SIDH attacks to be applicable. More precisely, the attacks are possible when $B^2 > 4A$, but an attacker could guess part of the isogeny so that the remaining part is short enough to be recovered through the SIDH attacks. This would suggest that if $2^{t'}B^2 \approx A$, an attacker can recover the unknown isogeny after iterating through $2^{t'}$ isogenies. This is the case for generic isogenies, but in the case of oriented ones, the attacker can brute force much longer isogenies at the same cost, since there are only limited options for any prime degree dividing A . In particular, after $2^{t'}$ computations, the attacker obtains isogenies of degree $A_{t'}$, the product of the t' largest primes dividing A . Thus, [Problem 3](#) is secure against the SIDH attacks when $A_{t'}B^2 \leq A$.

Assuming this condition is satisfied, **Problem 3** appears to be secure since the oriented-isogeny structure does not interact with the revealed torsion information, which does not make the problem easier. Lastly, before the attacks by Castryck and Decru, Maino, Martindale, Panny, Pope and Wesolowski, and Robert, SIDH with unbalanced parameters was vulnerable to torsion-point attacks [40,42] that relied on knowledge of the endomorphism ring of the starting curve. These attacks similarly do not apply to **Problem 3** since the torsion information is much lower than what is needed.

3.4 On the choice of the artificial orientation

As mentioned earlier in this section, a recent analysis [12] has shown it is possible to solve **Problem 1** in some particular cases:

- When the starting curve E_0 is defined over \mathbb{F}_p and the corresponding orientation (G_1, G_2) is such that G_1, G_2 or both are fixed by the Frobenius endomorphism.
- When the curve E_0 is not defined over \mathbb{F}_p , the attacks also extends to the case where the curve E_0 and its Frobenius conjugate $E_0^{(p)}$ are connected by a short isogeny $\psi : E_0 \rightarrow E_0^{(p)}$ and the artificial orientation (G_1, G_2) is such that G_1, G_2 or both are fixed by the endomorphism $\pi \circ \hat{\psi}$.
- When the starting curve admits a small endomorphism that fixes one or both groups G_1 and G_2 in a given artificial orientation.

We describe here another specific case where **Problem 1** is potentially easy to solve. Assume that E_0 is defined over \mathbb{F}_p (or it is close to its Frobenius conjugate; the case follows similarly) and fix an artificial orientation (G_1, G_2) where some subgroups of G_1 and G_2 are fixed by the Frobenius endomorphism. Let $\phi : E_0 \rightarrow E$ be a secret isogeny artificially oriented by (G_1, G_2) ; then, when the end curve E is also defined over \mathbb{F}_p , this indicates that the kernel of the secret isogeny is fixed by the Frobenius endomorphism, which discards any artificially oriented isogeny whose kernel is not fixed by the Frobenius. This means that one can discard several impossible secret keys just by looking at the field of definition of the end curve. In order to avoid this, either all of the end curves need to be defined over \mathbb{F}_p , or none of them can be defined over \mathbb{F}_p . In the first case, G_1 and G_2 are fixed by the Frobenius and the artificial orientation is the same as that of CSIDH. As highlighted above, this is not secure, hence the end curves should be defined over \mathbb{F}_{p^2} . To ensure this, no subgroup of the groups G_1 and G_2 should be fixed by the Frobenius. For any given prime ℓ , there are $\ell + 1$ cyclic groups of order ℓ , and at most two of them are fixed by the Frobenius endomorphism. Hence $\ell - 1$ cyclic groups of order ℓ are not fixed by the Frobenius, and these ones can be used in the artificial orientations. Note that that when $\ell = 2$, there may be only one group that is not fixed by the Frobenius: in such a case, the isogeny degrees A and B can be selected to be both odd. This issue with this approach is that there is no guarantee that none of the attacks from [12] listed earlier does not apply. For example, there is no guarantee that the groups in

the artificial orientations are not fixed by a small endomorphism (that could be exploited by an attacker). Therefore, all the strategies proposed till now do not provide enough insurance that the starting curve and the artificial orientations are secure.

In order to avoid all the problematic cases discussed above, we propose to use a uniformly random supersingular elliptic curve as a starting curve. Such a curve is defined over \mathbb{F}_{p^2} with overwhelming probability ($\approx 1 - \frac{1}{\sqrt{p}}$), it is far from \mathbb{F}_p curves, and it is not connected to its Frobenius conjugate by a short isogeny (with overwhelming probability). Such a curve can be generated by performing a long publicly-verifiable uniformly-random walk starting from a known supersingular elliptic curve. Practically, this walk can be computed by “nothing-up-my-sleeve” techniques: fix a seed s (say, the string “binSIDH+terSIDH”), compute its hash $h = \mathcal{H}(s)$ (where \mathcal{H} is a cryptographically secure hash function), and pass it as input to the CGL hash function [13]. The CGL output curve is then the starting curve E_0 of binSIDH and terSIDH. The artificial orientations are then generated as $\mathcal{A} = (\langle P_A \rangle, \langle Q_A \rangle)$ and $\mathcal{B} = (\langle P_B \rangle, \langle Q_B \rangle)$, where (P_A, Q_A) and (P_B, Q_B) are canonical bases of $E_0[A]$ and $E_0[B]$, respectively. Note that this approach is publicly verifiable by anyone and does not required any trusted setup.

In the rest of this paper, we assume that the starting curve E_0 and the artificial orientations are chosen this way.

4 The binSIDH and terSIDH protocols

In this section, we propose two new protocols: binSIDH and terSIDH. Both protocols translate the SIDH key exchange to the setting of artificially oriented curves and isogenies. The former restricts itself to fixed-degree isogenies, while the latter relies on variable-degree isogenies to improve on efficiency and compactness.

4.1 binSIDH

We first introduce binSIDH, which restricts itself to isogenies of full degree. The protocols rely on the fact that A -oriented curves provide sufficient information to compute parallel isogenies. More formally, let A be a product of t distinct primes $A = \prod_{i=1}^t p_i$ and write $A = A_1 A_2$ for a multiplicative splitting of A with $\gcd(A_1, A_2) = 1$. Then, given two A -oriented curves (E, \mathfrak{A}) and (E', \mathfrak{A}') connected by a B -isogeny $\phi : E \rightarrow E'$, where $\mathfrak{A} = (\langle G_1 \rangle, \langle G_2 \rangle)$ and $\mathfrak{A}' = (\langle G'_1 \rangle, \langle G'_2 \rangle)$, the isogenies

$$\psi : E \rightarrow E / \langle [A_1]G_1 + [A_2]G_2 \rangle, \quad \psi' : E' \rightarrow E' / \langle [A_1]G'_1 + [A_2]G'_2 \rangle$$

are parallel, i.e. we have $\ker \psi' = \phi(\ker \psi)$ and the codomain curves are also B -isogenous, connected by the isogeny ϕ' with kernel $\ker \phi' = \psi(\ker \phi)$.

The isogenies ψ and ψ' are thus determined by the splitting of A as $A = A_1 A_2$. In other words, if we represent the subgroups $\langle G_1 \rangle$ and $\langle G_2 \rangle$ as

$$\begin{aligned} \langle G_1 \rangle &= \langle G_1^1, G_1^2, \dots, G_1^t \rangle, \\ \langle G_2 \rangle &= \langle G_2^1, G_2^2, \dots, G_2^t \rangle, \end{aligned} \quad \text{where } \begin{cases} \text{ord}(G_1^i) = p_i, \\ \text{ord}(G_2^i) = p_i, \end{cases} \quad (1)$$

then the kernel of ψ is determined by selectively choosing either G_1^i or G_2^i to be in the kernel of ψ , for every $i \in [t]$. The same holds for the isogeny ψ' and the generators G_1' and G_2' . This suggests the following notation: if we fix an artificial A -orientation $(E, \mathfrak{A} = (G_1, G_2))$, where $A = \prod_{i=1}^t p_i$, we can associate a vector $\mathbf{a} \in \{1, 2\}^t$ to any \mathfrak{A} -oriented isogeny ϕ by writing

$$\ker \phi = \langle G_{\mathbf{a}_1}^1, G_{\mathbf{a}_2}^2, \dots, G_{\mathbf{a}_t}^t \rangle, \quad (2)$$

where the points G_1^i and G_2^i are defined as in Eq. (1) and \mathbf{a}_i denotes the i -th element of \mathbf{a} . Throughout the rest of the paper, we write $\langle \mathbf{a}, \mathfrak{A} \rangle$ to denote the subgroup corresponding to the orientation \mathfrak{A} with secret vector \mathbf{a} , as computed in Eq. (2).

We showed in Section 3.2 that we consider it secure to reveal artificially oriented curves since the SIDH attacks are inapplicable. Moreover, artificial orientations allow computations of parallel isogenies, and if the order A is sufficiently composite, the number of potential parallel isogenies is exponentially large. That is because the value A is the product of t distinct primes, which means there are 2^t potential splittings $A = A_1 A_2$. This suggests it is possible to replicate the SIDH key exchange with artificially oriented isogenies and to obtain a secure protocol that is immune to the SIDH attacks. We call the resulting construction binSIDH, and we represent it in Fig. 1.

Setup. Let λ be the security parameter and t an integer depending on λ . Let $p = ABf - 1$ be a prime such that $A = \prod_{i=1}^t \ell_i$ and $B = \prod_{i=1}^t q_i$ are coprime integers, ℓ_i, q_i are distinct small primes, $A \approx B \approx \sqrt{p}$ and f is a small cofactor. Let E_0 be a supersingular elliptic curve defined over \mathbb{F}_{p^2} with $\#E_0(\mathbb{F}_{p^2}) = (p+1)^2$. Let \mathfrak{A} be an artificial A -orientation on E_0 and let \mathfrak{B} be an artificial B -orientation on E_0 . The public parameters are $E_0, p, A, B, \mathfrak{A}$ and \mathfrak{B} .

KeyGen. Alice samples uniformly at random a vector \mathbf{a} from $\{1, 2\}^t$ and computes the \mathfrak{A} -oriented isogeny $\phi_A : E_0 \rightarrow E_A$ of degree A defined by \mathbf{a} . She also computes the push forward \mathfrak{B}' of \mathfrak{B} on E_A through ϕ_A . Her secret key is \mathbf{a} and her public key is (E_A, \mathfrak{B}') . Analogously, Bob samples uniformly at random a vector \mathbf{b} from $\{1, 2\}^t$ and computes the \mathfrak{B} -oriented isogeny $\phi_B : E_0 \rightarrow E_B$ of degree B defined by \mathbf{b} . He also computes the push forward \mathfrak{A}' of \mathfrak{A} on E_B through ϕ_B . His secret key is \mathbf{b} and his public key is (E_B, \mathfrak{A}') .

SharedKey. Upon receiving Bob’s public key (E_B, \mathfrak{A}') , Alice checks that \mathfrak{A}' is an artificial A -orientation on E_B , if not she aborts. She computes the \mathfrak{A}' -oriented isogeny $\phi'_A : E_B \rightarrow E_{BA}$ of degree A defined by \mathbf{a} . Her shared key is $j(E_{BA})$. Similarly, upon receiving (E_A, \mathfrak{B}') , Bob checks that \mathfrak{B}' is an artificial B -orientation on E_A , if not he aborts. He computes the \mathfrak{B}' -oriented isogeny $\phi'_B : E_A \rightarrow E_{AB}$ of degree B defined by \mathbf{b} . His shared key is $j(E_{AB})$.

Fig. 1: The binSIDH protocol.

4.2 The terSIDH variant

We now introduce **terSIDH**, a variant of **binSIDH** that is more efficient and more compact, but these improvements come at the cost of relying on variable-degree isogenies. In **binSIDH**, every A -oriented isogeny ϕ is determined by a binary choice for each prime p_i dividing A : the p_i -degree isogeny has kernel generated by either G_1^i or G_2^i . However, we can introduce a third option by allowing the isogeny to not have a p_i component. In other words, write ϕ as the composition of t isogenies $\phi = \phi_t \circ \dots \circ \phi_2 \circ \phi_1$; then, the isogeny ϕ_i has kernel generated by G_1^i , G_2^i , or \mathcal{O} . We thus extend the notation introduced in the previous section by letting the vector \mathbf{a} have entries in $\{0, 1, 2\}$, and we set $G_0^i = \mathcal{O}$ for all $i \in [t]$. The full protocol is described in [Fig. 2](#).

Compared to **binSIDH**, **terSIDH** introduces more choices for each prime p_i . In particular, it provides three choices, which means that every p_i dividing $p + 1$ provides $\log_2 3 \approx 1.6$ bits of security. Interestingly, **terSIDH** is the first countermeasure technique against the SIDH attacks that can provide more than one bit of security per prime p_i . This means that, to provide enough security, the isogeny degrees should be at least the product of $t \approx \lambda/1.6$ primes, and thus **terSIDH** can use significantly smaller parameters and shorter isogenies, leading to a more efficient and more compact protocol. However, to achieve this, we necessarily rely on variable-degree isogenies. This has some disadvantages: from an implementation perspective, the varying degree may make it harder to obtain constant-time implementations, as seen in the case of CSIDH implementations [\[2, 16\]](#). The other issue, as argued in [\[3, 7\]](#), is that it appears to be hard to construct zero-knowledge proofs of variable-degree isogenies because all known approaches invariably leak the secret isogeny degree. This causes a major issue in the development of proofs of **terSIDH** public key correctness, and it may prevent **terSIDH** from being an SIDH drop-in replacement for advanced constructions.

Setup. Let λ be the security parameter and t an integer depending on λ . Let $p = Abf - 1$ be a prime such that $A = \prod_{i=1}^t \ell_i$ and $B = \prod_{i=1}^t q_i$ are coprime

integers, ℓ_i, q_i are distinct small primes, $A \approx B \approx \sqrt{p}$ and f is a small cofactor. Let E_0 be a supersingular elliptic curve defined over \mathbb{F}_{p^2} with $\#E_0(\mathbb{F}_{p^2}) = (p+1)^2$. Let \mathfrak{A} be an artificial A -orientation on E_0 and let \mathfrak{B} be an artificial B -orientation on E_0 . The public parameters are $E_0, p, A, B, \mathfrak{A}$ and \mathfrak{B} .

KeyGen. Alice samples uniformly at random a vector \mathbf{a} from $\{0, 1, 2\}^t$ and computes the \mathfrak{A} -oriented isogeny $\phi_A : E_0 \rightarrow E_A$ defined by \mathbf{a} , whose degree divides A . She also computes the push forward \mathfrak{B}' of \mathfrak{B} on E_A through ϕ_A . Her secret key is \mathbf{a} and her public key is (E_A, \mathfrak{B}') . Analogously, Bob samples uniformly at random a vector \mathbf{b} from $\{0, 1, 2\}^t$ and computes the \mathfrak{B} -oriented isogeny $\phi_B : E_0 \rightarrow E_B$ defined by \mathbf{b} , whose degree divides B . He also computes the push forward \mathfrak{A}' of \mathfrak{A} on E_B through ϕ_B . His secret key is \mathbf{b} and his public key is (E_B, \mathfrak{A}') .

SharedKey. Upon receiving Bob's public key (E_B, \mathfrak{A}') , Alice checks that \mathfrak{A}' is an artificial A -orientation on E_B , if not she aborts. She computes the \mathfrak{A}' -oriented isogeny $\phi'_A : E_B \rightarrow E_{BA}$ of degree A defined by \mathbf{a} . Her shared key is $j(E_{BA})$. Similarly, upon receiving (E_A, \mathfrak{B}') , Bob checks that \mathfrak{B}' is an artificial B -orientation on E_A , if not he aborts. He computes the \mathfrak{B}' -oriented isogeny $\phi'_B : E_A \rightarrow E_{AB}$ of degree B defined by \mathbf{b} . His shared key is $j(E_{AB})$.

Fig. 2: The **terSIDH** protocol. This is nearly the same as Fig. 1, with the main difference being that **KeyGen** samples ternary secrets.

4.3 One more variant

It is possible to define a third variant of these protocols that relies on partial artificial orientations. Rather than revealing the images of two linearly independent points G_1 and G_2 , the protocol only reveals the image of one point G . Then, for each G^i of coprime order that make up G , the possible isogenies are computed by choosing whether G^i is in the kernel of the isogeny or not. Using the vector notation, its entries are chosen in $\{0, 1\}$.

Since the choice is binary, such a protocol would require similar parameters as **binSIDH**, while also having the disadvantages of variable-degree isogenies discussed in the context of **terSIDH**. As such, it does not appear to have any meaningful advantage over the proposed constructions. However, the information that is revealed about the secret isogeny is less: not only its degree remains unknown, as in **terSIDH**, but its action on a single cyclic group is revealed. This suggests that such a variant might be relevant if further cryptanalytic breakthroughs affect the security of **binSIDH** and **terSIDH**.

5 An oriented/non-oriented hybrid approach

There are applications where it is desirable for one party to be significantly more efficient than the other. For example, this is the case for resource-constrained devices communicating to powerful servers, but it also arises in advanced constructions: for instance, in oblivious pseudorandom function protocols, it is generally desired that the client is more efficient than the server. In this section, we propose a technique that allows us to introduce trade-offs between the two parties and enable one participant to obtain more efficient zero-knowledge proofs, which makes this approach more appealing for advanced protocols that requires proofs of isogeny knowledge. This technique has the added benefit of reducing the overall prime size for `binSIDH`, while the ternary variant has primes of comparable size as `terSIDH`.

In the previously presented protocols, both parties relied on artificially oriented isogenies to avoid the SIDH attacks. However, the artificial orientation also requires to use significantly longer isogenies than those used in the original SIDH protocol. This suggests that it may be possible to reveal some unscaled torsion information without affecting the security of the protocol, and if the isogeny is sufficiently long, the revealed torsion may be large enough to allow the computation of parallel isogenies that also guarantee sufficient security. In other words, we can build a secure protocol through a hybrid approach where one party computes `binSIDH`-like (or `terSIDH`-like) isogenies while the other party computes SIDH-like isogenies.

More formally, let Bob denote the party computing `binSIDH`-like isogenies, which means he computes artificially B -oriented isogenies where $B = \ell_1 \cdots \ell_n$; let Alice be the party computing SIDH-like isogenies of degree A , i.e. isogenies whose kernel is generated by $P_A + [\alpha]Q_A$, for some secret $\alpha \in \mathbb{Z}_A$ and fixed points P_A, Q_A . Fix a starting curve E_0 , points P_A, Q_A , and a B -orientation $\mathfrak{B} = (G_1, G_2)$, Alice's public key consists of the codomain of her secret isogeny, together with the image of \mathfrak{B} under her secret isogeny, while Bob's public key includes the codomain of his secret B -oriented isogeny, together with the images of P_A and Q_A .

Since Alice is computing SIDH-like isogenies, the degree of her secret isogeny can be very smooth (concretely, this will be a power of two); while this reduces the size of the isogeny degree of one party, the degree of the other party needs to increase to guarantee sufficient security. Thus, the resulting prime p is generally of comparable size to that used in `binSIDH` and `terSIDH`. With this setup, we can take A to be considerably smaller and smoother than B ; this means that Alice can be much more efficient in computing her isogenies. Not only that, but zero-knowledge proofs of knowledge of an A -isogeny, both ad-hoc [25] and generic [20], can be much more compact and efficient. More generally, computing SIDH-like isogenies allows one party to fully reuse the range of techniques developed for SIDH. The resulting schemes are described in Fig. 3.

Setup. Let λ be the security parameter and t an integer depending on λ . Let $p = ABf - 1$ be a prime such that $A = 2^a$ ($a \approx 2\lambda$) and $B = \prod_{i=1}^t \ell_i$ are coprime integers, ℓ_i are distinct small odd primes, and f is a small cofactor. Let E_0 be a supersingular elliptic curve defined over \mathbb{F}_{p^2} with $\#E_0(\mathbb{F}_{p^2}) = (p+1)^2$. Let \mathfrak{B} be an artificial B -orientation on E_0 and set $E_0[A] = \langle P_A, Q_A \rangle$. The public parameters are E_0, p, P_A, Q_A and \mathfrak{B} .

KeyGen (Alice). Alice samples uniformly at random an integer $\alpha \in \mathbb{Z}/A\mathbb{Z}$ and computes $\phi_A : E_0 \rightarrow E_A$ of kernel $\langle P_A + [\alpha]Q_A \rangle$. Her secret key is α and her public key is the artificially B -oriented curve $(E_A, \phi_A(\mathfrak{B}))$.

KeyGen (Bob). Bob samples uniformly at random a vector \mathbf{b} from $\{1, 2\}^t$ and computes the \mathfrak{B} -oriented isogeny $\phi_B : E_0 \rightarrow E_B$ of degree B defined by \mathbf{b} . His secret key is \mathbf{b} and his public key is $(E_B, \phi_B(P_A), \phi_B(Q_A))$.

SharedKey (Alice). Upon receiving Bob's public key (E_B, R, S) , Alice checks that $e_A(R, S) = e_A(P_A, Q_A)^B$, if not she aborts. She computes the isogeny $\phi'_A : E_B \rightarrow E_{BA}$ of kernel $\langle R + [\alpha]S \rangle$. Her shared key is $j(E_{BA})$.

SharedKey (Bob). Upon receiving (E_A, \mathfrak{B}') , Bob checks that \mathfrak{B}' is an artificial B -orientation on E_A , if not he aborts. He computes the \mathfrak{B}' -oriented isogeny $\phi'_B : E_A \rightarrow E_{AB}$ of degree B defined by \mathbf{b} . His shared key is $j(E_{AB})$.

Fig. 3: The $\text{binSIDH}^{\text{hyb}}$ protocol. A similar variant, based on terSIDH , can be obtained by changing Bob's KeyGen algorithm to sample vectors from $\{0, 1, 2\}^t$.

6 Security analysis

In this section, we analyze the security of the proposed protocols, both binSIDH and terSIDH , as well as their hybrid variants $\text{binSIDH}^{\text{hyb}}$ and $\text{terSIDH}^{\text{hyb}}$.

We analyzed the hardness assumptions relative to artificial orientations in [Section 3.2](#), which guarantees it is unfeasible for an attacker to recover a secret key from a public key. In particular, the hardness of [Problem 2](#) guarantees the security of binSIDH and terSIDH against key-recovery attacks, while the hardness of [Problem 1](#) and [3](#) protects $\text{binSIDH}^{\text{hyb}}$ and $\text{terSIDH}^{\text{hyb}}$ from key-recovery attacks ([Problem 1](#) for Alice's public key and [Problem 3](#) for Bob's). However, the security of the key-exchange protocols, as well as any other protocol built on those, depends on the hardness of a different problem, which we call the Artificially Oriented Computational Diffie-Hellman (AO-CDH) problem.

Problem 4 (AO-CDH). Let the notation be as in [Fig. 1](#). Let $\phi_A : E_0 \rightarrow E_A$ be a \mathfrak{A} -isogeny, and $\phi_B : E_0 \rightarrow E_B$ be a \mathfrak{B} -isogeny. Given $(E_A, \phi_A(\mathfrak{B}))$ and $(E_B, \phi_B(\mathfrak{A}))$, compute $j(E_{AB})$, where E_{AB} is the codomain of the push-forward of ϕ_A under ϕ_B (or vice versa).

The problem, as stated, guarantees the security of **terSIDH**. We can easily obtain similar problems for the remaining protocols by either requiring that the isogenies have fixed degrees (**binSIDH**) or allowing one party to use unoriented isogenies (**binSIDH^{hyb}**, **terSIDH^{hyb}**). We can also consider a decisional variant of these problems, where given an additional j -invariant j' , the problem asks to determine whether $j' = j(E_{AB})$. While the security of the proposed protocols does not depend on such decisional problems, advanced constructions based on these protocols might require such an assumption.

The relationship between these problems and those introduced in [Section 3.2](#) is similar to that between the Computational Diffie-Hellman problem and the Discrete Logarithm problem, or between the Supersingular Computational Diffie-Hellman problem and the Computational Supersingular Isogeny problem [\[35\]](#). While there exists no known reduction from the problems in [Section 3.2](#) to [Problem 4](#), it is likely that any attack that breaks the proposed protocols would need to efficiently solve the problems of [Section 3.2](#).

Remark 3. In **binSIDH** and **terSIDH**, the two parties reveal the codomain of their secret isogenies, together with only the images of two disjoint cyclic subgroups. This is, in some sense, optimal, as it is the minimum amount of information needed for the other party to compute the push-forwards. Thus, if any major cryptanalytic breakthrough managed to break **binSIDH** and **terSIDH**, it seems likely that any possible SIDH-like construction would equally be broken, including the existing countermeasures against the SIDH attacks [\[29\]](#).

6.1 The relation with the Uber-isogeny problem

When proposing **SÉTA** [\[24\]](#), its authors also introduced the Uber-isogeny problem, which is the following:

Problem 5 (D-Uber Isogeny Problem). Let $p > 3$ be a prime and let $\mathfrak{D} = \mathbb{Z}[w]$ be a quadratic order of discriminant Δ . Let E_0 and E be two \mathfrak{D} -oriented supersingular curves, and let $\theta \in \text{End}(E_0)$ be an endomorphism such that $\mathbb{Z}[\theta] \cong \mathfrak{D}$, that is θ allows to explicitly embed $\mathbb{Z}[\theta]$ into $\text{End}(E_0)$. Given E_0 , E and θ , find a power-smooth ideal \mathfrak{a} of norm co-prime with Δ such that $[\mathfrak{a}] \in \text{Cl}(\mathfrak{D})$ is such that $E \cong \mathfrak{a} * E_0$.

In other words, $\phi : E_0 \rightarrow E$ is an \mathfrak{D} -oriented isogeny, one is given the actual embedding of \mathfrak{D} into $\text{End}(E_0)$ but not that of \mathfrak{D} into $\text{End}(E)$, and one is asked to recover ϕ , or an equivalent isogeny of power-smooth degree. The authors of **SÉTA** [\[24\]](#) showed that the security of SIDH [\[35\]](#), CSIDH [\[11\]](#), OSIDH [\[19\]](#) and **SÉTA** [\[24\]](#) reduces to the Uber-isogeny problem. It is natural to wonder whether the security of our schemes can similarly be reduced to the Uber-isogeny problem as well. As we will show below, this is possible when the endomorphism ring of the starting curve E_0 is known, but the reduction is not trivial. In what follows, we relate [Problem 4](#) to the Uber-isogeny problem.

A first approach proceeds as follows. Let $\phi_A : E_0 \rightarrow E_A$ be Alice's artificially oriented isogeny with respect to an artificial orientation $\mathfrak{A} = (G_1, G_2)$. If the

endomorphism ring $\text{End}(E_0) \cong \mathcal{O}_0$ of the starting curve E_0 is known, then one can efficiently compute an endomorphism $\theta \in \text{End}(E_0)$ such that $\theta(G_1) = G_1$ and $\theta(G_2) = G_2$. Let w be a quaternion such that its norm and trace agree with θ , i.e. $w\bar{w} = N(w) = \deg \theta$ and $w + \bar{w} = \text{tr}(w) = \theta + \hat{\theta}$, and set $\mathfrak{D} = \mathbb{Z}[w]$. Then E_0 and E_A are \mathfrak{D} -oriented supersingular curves, and θ provides the actual embedding of \mathfrak{D} into $\text{End}(E_0)$. Any algorithm that solves the Uber-isogeny problem can be used to recover a power-smooth \mathfrak{D} -oriented isogeny $\psi_A : E_0 \rightarrow E_A$.

Such a reduction, however, presents an issue: with high probability, the recovered isogeny $\psi_A : E_0 \rightarrow E_A$ cannot be used as the secret in binSIDH . This is because Bob only reveals $\phi_B(\mathfrak{A})$, which allows the attacker to compute only the push-forwards of isogenies oriented by (G_1, G_2) and thus whose degree divides the order A . This implies that the knowledge of ψ_A is not sufficient to compute the shared secret in the key exchange. To solve this issue, we force both Alice's artificial orientation and Bob's artificial orientation to be restrictions of the same bigger \mathfrak{D} -orientation on E_0 .

As before, let us assume that the endomorphism ring $\text{End}(E_0) \cong \mathcal{O}_0$ of the starting curve E_0 is known. Let $\mathfrak{A} = (G_1^a, G_2^a)$ and $\mathfrak{B} = (G_1^b, G_2^b)$ be Alice's and Bob's artificial orientations, respectively. Set $G_1 = G_1^a \oplus G_1^b$ and $G_2 = G_2^a \oplus G_2^b$. It is possible to efficiently compute an endomorphism $\theta \in \text{End}(E_0)$ such that $\theta(G_1) = G_1$ and $\theta(G_2) = G_2$. Let w be a quaternion such that $w + \bar{w} = \text{tr}(w) = \theta + \hat{\theta}$ and $w\bar{w} = N(w) = \deg \theta$, and set $\mathfrak{D} = \mathbb{Z}[w]$. Then E_0 , E_A and E_B are all \mathfrak{D} -oriented supersingular elliptic curves, and θ provides the actual embedding of \mathfrak{D} into $\text{End}(E_0)$. Moreover, both Alice's artificially oriented secret isogeny $\phi_A : E_0 \rightarrow E_A$ and Bob's artificially oriented secret isogeny $\phi_B : E_0 \rightarrow E_B$ are \mathfrak{D} -oriented. Thus, the reduction starts by first using the algorithm to solve the Uber-isogeny problem with E_0 and E_A to recover a power-smooth \mathfrak{D} -oriented isogeny $\psi_A : E_0 \rightarrow E_A$, and then with E_0 and E_B to recover a power-smooth \mathfrak{D} -oriented isogeny $\psi_B : E_0 \rightarrow E_B$. Now, since all curves are \mathfrak{D} -oriented the underlying binSIDH scheme can be interpreted within the framework of the OSIDH [19] protocol. To obtain the shared secret, the reduction concludes by computing the push-forward ψ'_B of ψ_B through ψ_A (or vice-versa). The j -invariant of the codomain of ψ'_B is the shared key.

6.2 Adaptive security

SIDH has been known to be vulnerable to active adaptive attacks [33,30], i.e. attacks where the target has a long-term static key and the attacker is a participant of the key exchange. In this section, we show how the proposed protocols are unfortunately similarly vulnerable to adaptive attacks.

In $\text{binSIDH}^{\text{hyb}}$ and $\text{terSIDH}^{\text{hyb}}$, one party computes SIDH-like isogenies. As such, they are vulnerable to exactly the same attacks that SIDH is. We can thus focus on active attacks against oriented isogenies, which covers the remaining cases.

Let us assume Alice is the target party, while Bob plays the role of the attacker. For simplicity, let us also assume we are in the case of binSIDH , where Alice's secret is the binary vector $\mathbf{a} \in \{1, 2\}^t$. The case of terSIDH follows similarly.

Bob can use potentially malicious public keys and check whether both parties obtained the same shared secret. In other words, the attacker has access to the following oracle:

$$\mathcal{O}(E, \mathfrak{A}, j') = \begin{cases} true & \text{if } j(E/\langle \mathbf{a}, \mathfrak{A} \rangle) = j', \\ false & \text{otherwise.} \end{cases}$$

Write A , the order of the artificial orientation \mathfrak{A} , as $A = \prod_{i=1}^t p_i$. To target the i -th bit of the secret key \mathbf{a}_i , the attacker can honestly compute the curve E_B and the image orientation $\mathfrak{A} = (G_1, G_2)$ and write $G_j = H_j^1 \oplus \dots \oplus H_j^t$ for $j \in \{1, 2\}$, where each H_j^k has order p_k , which are all pairwise coprime. Then, if I_1^i is any cyclic subgroup of order p_i such that $I_1^i \cap H_1^i = \{\mathcal{O}\}$, the attacker can define $\mathfrak{A}' = (G'_1, G_2)$, where G'_1 is the same subgroup as G_1 with H_1^i replaced by I_1^i , i.e. $G'_1 = H_1^1 \oplus \dots \oplus I_1^i \oplus \dots \oplus H_1^t$. The attacker can also obtain the j -invariant j_{AB} corresponding to the shared secret of an honest exchange and query the oracle $\mathcal{O}(E_B, \mathfrak{A}', j_{AB})$. If the oracle returns *true*, the shared secret is unchanged: this means that modified subgroup did not affect the computations, and thus $\mathbf{a}_i = 2$. Otherwise, the modified subgroup did change the shared secret, and thus $\mathbf{a}_i = 1$.

The active attack against the proposed protocols is slightly more powerful than the GPST attack. It does not involve carefully crafted torsion points, and it allows to target any bit of the secret key without necessarily proceeding in order. In the PKE setting, one party can achieve long-term security with the use of the Fujisaki-Okamoto transform [31], while in the key exchange setting, it is possible to obtain active security for both parties, thus obtaining a non-interactive key exchange, by introducing a proof of public key correctness. For artificially oriented curves, this can be achieved by adapting the zero-knowledge proof of masked public keys from [3] to work with independently scaled points.

7 Implementation

7.1 Parameter selection

Following the security analysis of Section 3.2, we generated parameter sets for the four proposed protocols at security levels $\lambda \in \{128, 192, 256\}$.

In `binSIDH` and `terSIDH`, both parties rely on oriented isogenies, and thus the degrees corresponding to both isogenies need to be quite large: in the case of `binSIDH`, at least the product of λ distinct primes. This is reduced to $\lambda/\log_2(3)$ for `terSIDH`, since each prime provides $\log_2(3)$ bits of security. To obtain a balanced trade-off between the two parties, we assign consecutive primes to different parties; in other words, the degree of Alice's isogenies is the product of t even-index primes, while the degree of Bob's isogenies is the product of t odd-index primes. Moreover, the isogeny degrees need to be coprime, and thus the underlying prime necessarily needs to be larger than the product of the first 2λ in `binSIDH` ($2\lambda/\log_2(3)$ in `terSIDH`). The resulting parameter sets for `binSIDH`

and **terSIDH** are summarized in **Table 1**, where we also list the corresponding public key sizes.

Remark 4 (Public-key compression). As in SIDH, public keys can be compressed by expressing the torsion points with respect to a deterministically generated basis [21]. This requires three coefficients in SIDH since both points can be scaled by the same value without affecting the SIDH computations, which means that one of the four coefficients can be fixed to one. In our case, however, the two points that generate artificial orientations can be scaled independently: this means that the public keys of the proposed protocols can be compressed to only two coefficients.

	λ	$\log p$	Alice				Bob			
			t	\mathcal{B}	$ \mathbf{pk} $	$ \mathbf{pk}_{\text{cmp}} $	t	B	$ \mathbf{pk} $	$ \mathbf{pk}_{\text{cmp}} $
binSIDH	128	2421	134	2^{11}	1816	907	134	2^{11}	1816	909
	192	3710	192	2^{12}	2783	1390	192	2^{12}	2783	1392
	256	5201	256	2^{12}	3901	1949	256	2^{12}	3901	1950
terSIDH	128	1568	93	2^{11}	1176	587	93	2^{11}	1176	588
	192	2295	128	2^{11}	1722	860	128	2^{11}	1722	861
	256	3035	162	2^{12}	2277	1137	162	2^{12}	2277	1139
binSIDH^{hyb}	128	2004	1	2	1503	937	203	2^{11}	1503	565
	192	3126	1	2	2345	1465	296	2^{11}	2345	878
	256	4267	1	2	3201	2004	387	2^{12}	3201	1195
terSIDH^{hyb}	128	1532	1	2	1149	701	156	2^{10}	1149	447
	192	2373	1	2	1780	1089	226	2^{11}	1780	690
	256	3216	1	2	2412	1479	293	2^{11}	2412	932

Table 1: Parameters for **binSIDH** and **terSIDH**. The column t reports the number of distinct primes dividing the degrees of Alice’s and Bob’s isogenies, while their smoothness bound is reported in the \mathcal{B} column. The columns $|\mathbf{pk}|$ and $|\mathbf{pk}_{\text{cmp}}|$ reports the size of the public keys of both parties, respectively uncompressed and compressed.

The size of the primes and public keys of **binSIDH** and **terSIDH** is a stark improvement over those of the existing countermeasures M-SIDH and MD-SIDH [29]. For instance, at $\lambda = 128$, the primes of **binSIDH** and **terSIDH** are $2.5\times$ and $8.8\times$ smaller than those in M-SIDH and MD-SIDH, respectively.⁵ While **terSIDH^{hyb}** requires larger parameters than **terSIDH**, **binSIDH^{hyb}** manages to

⁵ Interestingly, in the **terSIDH** case, the variable-degree isogenies allow us to achieve smaller parameters, while in MD-SIDH, the variable-degree isogenies require larger parameters because of the information leakage due to pairing computations.

be smaller and more efficient than binSIDH. When compared to M-SIDH, the underlying prime in binSIDH^{hyb} is $2.9\times$ smaller.

7.2 Implementation results

We developed a proof-of-concept implementation of all four protocols in SageMath [45], based on the Kummer Line library [41] to estimate the running times of the proposed protocols⁶. We report the average running times on an Apple M1 PRO CPU in Table 2.

	λ	$\log p$	Timings (s)			
			KeyGen _A	KeyGen _B	SharedKey _A	SharedKey _B
binSIDH	128	2421	13.69	13.86	9.40	9.46
	192	3710	48.69	49.36	27.39	27.81
	256	5201	140.79	140.57	94.13	95.67
terSIDH	128	1570	2.07	2.09	1.38	1.38
	192	2297	6.84	6.83	4.50	4.39
	256	3039	15.68	16.03	10.00	10.35
binSIDH ^{hyb}	128	2004	0.23	14.33	0.22	10.66
	192	3126	0.62	56.77	0.61	42.85
	256	4267	1.41	157.58	1.34	117.07
terSIDH ^{hyb}	128	1532	0.16	3.21	0.16	1.96
	192	2373	0.47	13.44	0.44	10.01
	256	3216	0.94	34.66	0.90	23.57

Table 2: Execution times in seconds of the SageMath proof-of-concept implementation. Since it is a PoC in a high-level language, we expect an optimized implementation of the same protocols to be several times more efficient.

The results of Table 2 show that the ternary variants significantly outperforms the binary ones, especially at higher security levels. This is because binSIDH uses larger prime fields and larger-degree isogenies than terSIDH.⁷ Moreover, terSIDH does not need to compute full-degree isogenies due to its varying-degree nature: it is thus likely that the benefits of this are reduced in a constant-time optimization. Nonetheless, the results of terSIDH are encouraging. At security $\lambda = 128$, the

⁶ The source code is available at <https://github.com/binary-ternarySIDH/bin-terSIDH-SageMath>

⁷ The specific SageMath implementation of VéluSqrt [6] that we rely on does not outperform Vélu’s formulae [47] until the isogeny degree is extremely large. We thus expect a low-level implementation to significantly improve the computation times of high-degree isogenies, more so than for lower-degree ones.

SharedKey computations take around 1.4 seconds, while key generation (which is run less often) requires about two seconds. The current implementation is only a proof of concept in a high-level language: we can thus expect it to be several times faster once optimally implemented in a low-level language.

Despite the lack of optimizations, the current implementation already outperforms optimized implementations of CSIDH with parameters sufficiently large to guarantee post-quantum security [16], which require between 2.8 and 5.8 seconds to compute a group action at security level one.⁸ Very recently, a new implementation of CSIDH [8] achieves lower running times for a single group action, which takes between 0.9 and 4.6 seconds. While the CSIDH implementation with a smaller prime outperforms the SageMath implementation of $\text{terSIDH}^{\text{hyb}}$, the former is heavily optimized: we thus expect a similarly optimized implementation $\text{terSIDH}^{\text{hyb}}$ to be significantly more efficient than CSIDH.

Comparing to other protocols, our proof-of-concept implementation outperforms the PoC implementation of FESTA [5], which is based on the same SageMath library and takes 3.5 seconds to encrypt and 10.1 seconds to decrypt. It is thus mostly likely that terSIDH provides the most efficient key exchange and encryption protocol among all isogeny-based protocols.

Moreover, the results of the hybrid variants show that it is possible to have very low running times for one party, at the cost of a slight increase in the running times of the other party. The hybrid variants significantly reduce the overall running time of a complete key exchange.

8 Conclusion

In this work, we introduced artificial orientations, and proposed two new protocols, binSIDH and terSIDH , that translate the SIDH key exchange to the artificially oriented isogeny setting. This allows us to develop two protocols that are resistant against the SIDH attacks, while also achieving significantly smaller parameters than the previously proposed countermeasures. We also proposed $\text{binSIDH}^{\text{hyb}}$ and $\text{terSIDH}^{\text{hyb}}$, hybrid variants of binSIDH and terSIDH respectively, that allow one party to have very short and efficient isogenies. To validate the concrete efficiency of the protocols, we developed a proof-of-concept implementation. Despite being far from optimal, it already outperforms existing implementations of other isogeny-based encryption protocols (both key exchanges and public-key encryption protocols), which suggests that optimized implementations of terSIDH and its hybrid variant might have practical running times.

In future work, we are interested in developing efficient and optimized implementations of binSIDH and terSIDH to accurately measure their running times. Moreover, this work opens up new possibilities that were previously closed by the SIDH attacks. In particular, it is interesting to assess the impact of the proposed protocols on the SIDH-based constructions, such as the round-optimal

⁸ Note, however, that the CSIDH implementations are constant-time, and that CSIDH does not require the Fujisaki-Okamoto [32] to obtain IND-CCA security.

OPRF construction by Basso [3], where we expect binSIDH and terSIDH to have a significant impact in reducing prime size and computational costs.

Acknowledgements. We would like to express our gratitude to the anonymous reviewers of ASIACRYPT 2023 for their valuable comments that helped improve this paper. We thank Wouter Castryck and Fre Vercauteren for sharing their early draft on attacks on some instances of M-SIDH and FESTA, the attacks described in this draft were useful in the security analysis of our schemes. The first author has been supported in part by EPSRC via grant EP/R012288/1, under the RISE (<http://www.ukrise.org>) programme.

References

1. Adj, G., Cervantes-Vázquez, D., Chi-Domínguez, J.J., Menezes, A., Rodríguez-Henríquez, F.: On the cost of computing isogenies between supersingular elliptic curves. In: Cid, C., Jacobson Jr., M.J. (eds.) SAC 2018. LNCS, vol. 11349, pp. 322–343. Springer, Heidelberg (Aug 2019). https://doi.org/10.1007/978-3-030-10970-7_15
2. Banegas, G., Bernstein, D.J., Campos, F., Chou, T., Lange, T., Meyer, M., Smith, B., Sotáková, J.: CTIDH: faster constant-time CSIDH. IACR TCHES **2021**(4), 351–387 (2021). <https://doi.org/10.46586/tches.v2021.i4.351-387>, <https://tches.iacr.org/index.php/TCHES/article/view/9069>
3. Basso, A.: A post-quantum round-optimal oblivious PRF from isogenies. Cryptology ePrint Archive, Report 2023/225 (2023), <https://eprint.iacr.org/2023/225>
4. Basso, A., Kutas, P., Merz, S.P., Petit, C., Sanso, A.: Cryptanalysis of an oblivious PRF from supersingular isogenies. In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021, Part I. LNCS, vol. 13090, pp. 160–184. Springer, Heidelberg (Dec 2021). https://doi.org/10.1007/978-3-030-92062-3_6
5. Basso, A., Maino, L., Pope, G.: FESTA: Fast Encryption from Supersingular Torsion Attacks. Cryptology ePrint Archive, Paper 2023/660 (2023), <https://eprint.iacr.org/2023/660>
6. Bernstein, D.J., De Feo, L., Leroux, A., Smith, B.: Faster computation of isogenies of large prime degree. Open Book Series **4**(1), 39–55 (2020). <https://doi.org/10.2140/obs.2020.4.39>
7. Beullens, W., Feo, L.D., Galbraith, S.D., Petit, C.: Proving knowledge of isogenies – a survey. Cryptology ePrint Archive, Paper 2023/671 (2023), <https://eprint.iacr.org/2023/671>, <https://eprint.iacr.org/2023/671>
8. Campos, F., Chavez-Saab, J., Chi-Domínguez, J.J., Meyer, M., Reijnders, K., Rodríguez-Henríquez, F., Schwabe, P., Wiggers, T.: On the practicality of post-quantum tls using large-parameter csidh. Cryptology ePrint Archive, Paper 2023/793 (2023), <https://eprint.iacr.org/2023/793>, <https://eprint.iacr.org/2023/793>
9. Castryck, W., Decru, T.: An Efficient Key Recovery Attack on SIDH. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023. pp. 423–447. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-30589-4_15
10. Castryck, W., Houben, M., Merz, S.P., Mula, M., van Buuren, S., Vercauteren, F.: Weak instances of class group action based cryptography via self-pairings. Cryptology ePrint Archive, Paper 2023/549 (2023), <https://eprint.iacr.org/2023/549>, <https://eprint.iacr.org/2023/549>

11. Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: CSIDH: An efficient post-quantum commutative group action. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018, Part III. LNCS, vol. 11274, pp. 395–427. Springer, Heidelberg (Dec 2018). https://doi.org/10.1007/978-3-030-03332-3_15
12. Castryck, W., Vercauteren, F.: A polynomial time attack on instances of M-SIDH and FESTA. To appear in ASIACRYPT 2023 (2023)
13. Charles, D.X., Lauter, K.E., Goren, E.Z.: Cryptographic hash functions from expander graphs. *Journal of Cryptology* **22**(1), 93–113 (Jan 2009). <https://doi.org/10.1007/s00145-007-9002-x>
14. Chen, M., Imran, M., Ivanyos, G., Kutas, P., Leroux, A., Petit, C.: Hidden stabilizers, the isogeny to endomorphism ring problem and the cryptanalysis of psidh (2023)
15. Childs, A.M., Jao, D., Soukharev, V.: Constructing elliptic curve isogenies in quantum subexponential time. *J. Math. Cryptol.* **8**(1), 1–29 (2014). <https://doi.org/10.1515/jmc-2012-0016>, <https://doi.org/10.1515/jmc-2012-0016>
16. Chávez-Saab, J., Chi-Domínguez, J.J., Jaques, S., Rodríguez-Henríquez, F.: The SQALE of CSIDH: sublinear Vélú quantum-resistant isogeny action with low exponents. *Journal of Cryptographic Engineering* **12**(3), 349–368 (Sep 2022). <https://doi.org/10.1007/s13389-021-00271-w>
17. Codogni, G., Lido, G.: Spectral theory of isogeny graphs (2023)
18. Colò, L., Kohel, D.: Orienting supersingular isogeny graphs. *Cryptology ePrint Archive, Report 2020/985* (2020), <https://eprint.iacr.org/2020/985>
19. Colò, L., Kohel, D.: Orienting supersingular isogeny graphs. *Journal of Mathematical Cryptology* **14**(1), 414–437 (2020)
20. Cong, K., Lai, Y.F., Levin, S.: Efficient isogeny proofs using generic techniques. *Cryptology ePrint Archive, Report 2023/037* (2023), <https://eprint.iacr.org/2023/037>
21. Costello, C., Jao, D., Longa, P., Naehrig, M., Renes, J., Urbanik, D.: Efficient compression of SIDH public keys. In: Coron, J.S., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part I. LNCS, vol. 10210, pp. 679–706. Springer, Heidelberg (Apr / May 2017). https://doi.org/10.1007/978-3-319-56620-7_24
22. Couveignes, J.M.: Hard homogeneous spaces. *Cryptology ePrint Archive, Report 2006/291* (2006), <https://eprint.iacr.org/2006/291>
23. Dartois, P., De Feo, L.: On the security of OSIDH. *Cryptology ePrint Archive, Report 2021/1681* (2021), <https://eprint.iacr.org/2021/1681>
24. De Feo, L., de Saint Guilhem, C., Fouotsa, T.B., Kutas, P., Leroux, A., Petit, C., Silva, J., Wesolowski, B.: Seta: Supersingular encryption from torsion attacks. In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021, Part IV. LNCS, vol. 13093, pp. 249–278. Springer, Heidelberg (Dec 2021). https://doi.org/10.1007/978-3-030-92068-5_9
25. De Feo, L., Dobson, S., Galbraith, S.D., Zobernig, L.: SIDH proof of knowledge. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022, Part II. LNCS, vol. 13792, pp. 310–339. Springer, Heidelberg (Dec 2022). https://doi.org/10.1007/978-3-031-22966-4_11
26. De Feo, L., Fouotsa, T.B., Kutas, P., Leroux, A., Merz, S.P., Panny, L., Wesolowski, B.: SCALLOP: Scaling the CSI-FiSh. In: Boldyreva, A., Kolesnikov, V. (eds.) *Public-Key Cryptography – PKC 2023*. pp. 345–375. Springer Nature Switzerland, Cham (2023)
27. De Feo, L., Kohel, D., Leroux, A., Petit, C., Wesolowski, B.: SQISign: Compact post-quantum signatures from quaternions and isogenies. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part I. LNCS, vol. 12491, pp. 64–93. Springer, Heidelberg (Dec 2020). https://doi.org/10.1007/978-3-030-64837-4_3

28. Delfs, C., Galbraith, S.D.: Computing isogenies between supersingular elliptic curves over \mathbb{F}_p . *Designs, Codes and Cryptography* **78**(2), 425–440 (Feb 2016). <https://doi.org/10.1007/s10623-014-0010-1>, <https://doi.org/10.1007/s10623-014-0010-1>
29. Fouotsa, T.B., Moriya, T., Petit, C.: M-SIDH and MD-SIDH: Countering SIDH Attacks by Masking Information. In: Hazay, C., Stam, M. (eds.) *Advances in Cryptology – EUROCRYPT 2023*. pp. 282–309. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-30589-4_10
30. Fouotsa, T.B., Petit, C.: A new adaptive attack on SIDH. In: Galbraith, S.D. (ed.) *CT-RSA 2022*. LNCS, vol. 13161, pp. 322–344. Springer, Heidelberg (Mar 2022). https://doi.org/10.1007/978-3-030-95312-6_14
31. Fujisaki, E., Okamoto, T.: Statistical zero knowledge protocols to prove modular polynomial relations. In: Kaliski Jr., B.S. (ed.) *CRYPTO'97*. LNCS, vol. 1294, pp. 16–30. Springer, Heidelberg (Aug 1997). <https://doi.org/10.1007/BFb0052225>
32. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Wiener, M.J. (ed.) *CRYPTO'99*. LNCS, vol. 1666, pp. 537–554. Springer, Heidelberg (Aug 1999). https://doi.org/10.1007/3-540-48405-1_34
33. Galbraith, S.D., Petit, C., Shani, B., Ti, Y.B.: On the security of supersingular isogeny cryptosystems. In: Cheon, J.H., Takagi, T. (eds.) *ASIACRYPT 2016, Part I*. LNCS, vol. 10031, pp. 63–91. Springer, Heidelberg (Dec 2016). https://doi.org/10.1007/978-3-662-53887-6_3
34. Jao, D., Azarderakhsh, R., Campagna, M., Costello, C., De Feo, L., Hess, B., Jalali, A., Koziel, B., LaMacchia, B., Longa, P., Naehrig, M., Renes, J., Soukharev, V., Urbanik, D., Pereira, G., Karabina, K., Hutchinson, A.: SIKE. Tech. rep., National Institute of Standards and Technology (2020), available at <https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions>
35. Jao, D., De Feo, L.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In: Yang, B.Y. (ed.) *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011*. pp. 19–34. Springer, Heidelberg (Nov / Dec 2011). https://doi.org/10.1007/978-3-642-25405-5_2
36. Kani, E.: The number of curves of genus two with elliptic differentials. *Journal für die reine und angewandte Mathematik* **1997**(485), 93–122 (1997). <https://doi.org/doi:10.1515/crll.1997.485.93>
37. Leroux, A.: A new isogeny representation and applications to cryptography. In: Agrawal, S., Lin, D. (eds.) *ASIACRYPT 2022, Part II*. LNCS, vol. 13792, pp. 3–35. Springer, Heidelberg (Dec 2022). https://doi.org/10.1007/978-3-031-22966-4_1
38. Maino, L., Martindale, C., Panny, L., Pope, G., Wesolowski, B.: A Direct Key Recovery Attack on SIDH. In: Hazay, C., Stam, M. (eds.) *Advances in Cryptology – EUROCRYPT 2023*. pp. 448–471. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-30589-4_16
39. Peikert, C.: He gives C-sieves on the CSIDH. In: Canteaut, A., Ishai, Y. (eds.) *EUROCRYPT 2020, Part II*. LNCS, vol. 12106, pp. 463–492. Springer, Heidelberg (May 2020). https://doi.org/10.1007/978-3-030-45724-2_16
40. Petit, C.: Faster algorithms for isogeny problems using torsion point images. In: Takagi, T., Peyrin, T. (eds.) *ASIACRYPT 2017, Part II*. LNCS, vol. 10625, pp. 330–353. Springer, Heidelberg (Dec 2017). https://doi.org/10.1007/978-3-319-70697-9_12
41. Pope, G.: Kummer Isogeny SageMath Library. <https://github.com/jack4818/KummerIsogeny> (2023)

42. de Quehen, V., Kutas, P., Leonardi, C., Martindale, C., Panny, L., Petit, C., Stange, K.E.: Improved torsion-point attacks on SIDH variants. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part III. LNCS, vol. 12827, pp. 432–470. Springer, Heidelberg, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84252-9_15
43. Robert, D.: Breaking SIDH in Polynomial Time. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023. pp. 472–503. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-30589-4_17
44. Silverman, J.H.: The Arithmetic of Elliptic Curves, vol. 106. Springer Science & Business Media (2009)
45. The Sage Developers: SageMath, the Sage Mathematics Software System (Version 9.8) (2023), <https://www.sagemath.org>
46. van Oorschot, P.C., Wiener, M.J.: Parallel collision search with cryptanalytic applications. *Journal of Cryptology* **12**(1), 1–28 (Jan 1999). <https://doi.org/10.1007/PL00003816>
47. Vélú, J.: Isogénies entre courbes elliptiques. *CR Acad. Sci. Paris, Séries A* **273**, 305–347 (1971)