SCALLOP: scaling the CSI-FiSh

Luca De Feo[1][0000−0002−9321−0773], Tako Boris Fouotsa[2][0000−0003−1821−8406], Péter Kutas[3,4], Antonin Leroux[5,11,12,13], Simon-Philipp Merz[6], Lorenz Panny[7], and Benjamin Wesolowski[8,9,10][0000−0003−1249−6077]

[1] IBM Research Europe, Zürich, Switzerland scallop@defeo.lu
[2] EPFL, Lausanne, Switzerland tako.fouotsa@epfl.ch
[3] Eötvös Loránd University, Budapest, Hungary
[4] University of Birmingham, UK p.kutas@bham.ac.uk
[5] DGA-MI, Bruz, France antonin.leroux@polytechnique.org
[6] Royal Holloway, University of London, Egham, UK research@simon-philipp.com
[7] Academia Sinica, Taipei, Taiwan lorenz@yx7.cc
[8] Univ. Bordeaux, CNRS, Bordeaux INP, IMB, UMR 5251, F-33400, Talence, France
[9] INRIA, IMB, UMR 5251, F-33400, Talence, France
[10] ENS de Lyon, CNRS, UMPA, UMR 5669, Lyon, France
[11] IRMAR, UMR 6625, Université de Rennes, France
[12] LIX, CNRS, Ecole Polytechnique, Institut Polytechnique de Paris
[13] INRIA

Abstract. We present SCALLOP: SCALable isogeny action based on Oriented supersingular curves with Prime conductor, a new group action based on isogenies of supersingular curves. Similarly to CSIDH and OSIDH, we use the group action of an imaginary quadratic order's class group on the set of oriented supersingular curves. Compared to CSIDH, the main benefit of our construction is that it is easy to compute the class-group structure; this data is required to uniquely represent — and efficiently act by — arbitrary group elements, which is a requirement in, e.g., the CSI-FiSh signature scheme by Beullens, Kleinjung and Vercauteren. The index-calculus algorithm used in CSI-FiSh to compute the class-group structure has complexity $L(1/2)$, ruling out class groups much larger than CSIDH-512, a limitation that is particularly problematic in light of the ongoing debate regarding the quantum security of cryptographic group actions.

Hoping to solve this issue, we consider the class group of a quadratic order of large prime conductor inside an imaginary quadratic field of small discriminant. This family of quadratic orders lets us easily determine the size of the class group, and, by carefully choosing the conductor,

* Author list in alphabetical order; see https://ams.org/profession/leaders/CultureStatement04.pdf. This research was funded in part by the EPSRC under grants EP/S01361X/1 and EP/P009301/1, the National Research, Development and Innovation Office within the Quantum Information National Laboratory of Hungary, the János Bolyai Research Scholarship of the Hungarian Academy of Sciences and by the ÚNKP-22-5 New National Excellence Program, the Academia Sinica Investigator Award AS-IA-109-M01, the Agence Nationale de la Recherche under grant ANR MELODIA (ANR-20-CE40-0013), and the France 2030 program under grant agreement No. ANR-22-PETQ-0008 PQ-TLS. Date of this document: 2023-10-23.

even exercise significant control on it — in particular supporting highly smooth choices. Although evaluating the resulting group action still has subexponential asymptotic complexity, a careful choice of parameters leads to a practical speedup that we demonstrate in practice for a security level equivalent to CSIDH-1024, a parameter currently firmly out of reach of index-calculus-based methods. However, our implementation takes 35 seconds (resp. 12.5 minutes) for a single group-action evaluation at a CSIDH-512-equivalent (resp. CSIDH-1024-equivalent) security level, showing that, while feasible, the SCALLOP group action does not achieve realistically usable performance yet.

1 Introduction

Isogeny-based cryptography was first proposed by Couveignes [21] in 1996, but not published at the time. The same idea was independently rediscovered by Rostovtsev and Stolbunov later [49]. Couveignes and Rostovtsev–Stolbunov (CRS) suggested a post-quantum key exchange based on the group action of an ideal class group on a class of ordinary elliptic curves. However, this scheme is very slow.

A major breakthrough for isogeny-based group actions was the invention of CSIDH [13]. The construction follows a similar blueprint as the CRS key exchange but the class group of an imaginary quadratic order acts on the set of supersingular elliptic curves defined over a prime field, rather, and this makes the scheme a lot faster for various reasons. CSIDH was the first efficient post-quantum action and its efficient public-key validation gives rise to non-interactive key exchange. While it is well known that CSIDH, like CRS, is susceptible to a quantum subexponential attack, the concrete size of parameters to achieve a certain security level has been a matter of debate [47,10,16].

Colò and Kohel generalized CSIDH-like schemes to obtain the "Oriented Supersingular Isogeny Diffie–Hellman" (OSIDH) key exchange [19], introducing the notion of orientations to handle the action of a generic class group on a set of supersingular curves. Since then, the OSIDH key exchange has been broken for the suggested parameters [23], but its generalisation of CSIDH remains a useful framework.

The first attempt to build isogeny-based signatures was outlined in Stolbunov's PhD thesis, where the Fiat–Shamir transform is applied to a Σ-protocol [51]. However, to instantiate the scheme it would be necessary to sample uniformly from the acting class group and, crucially, to compute a canonical representative for each class group element efficiently. The first requirement could be approximated sufficiently well, but the second one remained elusive. Instead of using canonical representatives, De Feo and Galbraith proposed the signature scheme SeaSign [25] which uses an abundantly redundant representation together with rejection sampling to make the distribution of class group elements independent from the secret key. Though it provides short signatures, signing time is still impractical for the fastest parameter set (2 minutes), even after further optimisations by Decru, Panny and Vercauteren [29].

Computing the class group structure of the acting group solves both challenges left to instantiate Stolbunov's signature scheme. By providing a simple canonical representation for class group elements, it also gives an easy way to sample uniformly, instead of resorting to expensive statistical methods. In 2019, Beullens, Kleinjung and Vercauteren [9] conducted a record breaking class group computation to find the class group structure and relation lattice of the class group of the imaginary quadratic field corresponding to the smallest CSIDH parameter set, CSIDH-512. This let them efficiently instantiate Stolbunov's signature, leading to CSI-FiSh [9]. CSI-FiSh is very efficient and is a building block for many other schemes such as threshold signatures [28,22,11], ring signatures [8,39]) and group signatures [7,18]. Furthermore, it is a basis for other primitives such as updatable encryption [41].

Unfortunately, the best known algorithms to compute the class group structure have complexity $L_{\Delta}(1/2)$, using the classic L-notation

$$L_x(\alpha) = \exp\left(O\left((\log x)^{\alpha}(\log\log x)^{1-\alpha}\right)\right),$$

where Δ denotes the discriminant of the number field. Instantiating CSI-FiSh for larger security levels of CSIDH would require class group computations that are currently out of reach. Yet, especially in light of recent debate about CSIDH's concrete quantum security, it is desirable to have an efficient isogeny-based signature scheme (and all the aforementioned primitives) at higher security levels.

This motivates the search for other isogeny actions that have better control on the class group, it is thus natural to look at orientations different from the one in CSIDH. However, choosing an orientation poses several challenges. First, it is usually hard to compute an orientation even if one knows that the curve is oriented by a particular order as discussed in [24]. Secondly, disclosing the orientation in the public key requires an efficient representation of the orientation. Then, the resulting group action should be efficiently computable. Finally, for a general orientation it is unclear how the structure of the class group can be computed, whereas special orientations may not lead to cryptographically secure group actions (see [19,23] and [2, Theorem 11.4]).

1.1 Contribution

We present SCALLOP, a new isogeny-based group action on supersingular curves. Following a standard approach [13,19], we use the group action of the class group of an imaginary quadratic order on a set of oriented supersingular curves. In an attempt to solve the scaling issue of CSI-FiSh, we explore the situation where the quadratic order $\mathfrak{O}$ of discriminant Δ has a large prime conductor f inside an imaginary quadratic field of very small discriminant d_0, i.e. $\Delta = f^2 d_0$. There are exact formulas and results to compute the structure of the class group in this case.

To make the computation of the resulting group action efficient, we study how to obtain effective and (hopefully) secure $\mathfrak{O}$-orientations for a generic quadratic order $\mathfrak{O}$, something known only in the special case of CSIDH, where $\mathfrak{O} = \mathbb{Z}[\sqrt{-p}]$,

prior to our work. In particular, we introduce a generic framework to evaluate the group action when $\mathfrak{O}$ contains a generator α such that the principal ideal $\mathfrak{O}\alpha$ can be factored as $\mathfrak{L}_1^2\mathfrak{L}_2$ for two ideals $\mathfrak{L}_1, \mathfrak{L}_2$ of smooth coprime norm. We then show how to instantiate this framework when $\mathfrak{O}$ is an order of large prime conductor and we provide an algorithm to perform the computation as efficiently as possible in this context. In particular, we provide a way to choose the conductor such that $\mathfrak{O}$ has a generator α of the correct form with essentially optimal size. As is customary in isogeny-based cryptography, this setup also requires to carefully select the characteristic of the finite field $\mathbb{F}_p$ for an efficient evaluation of the group action.

To generate concrete parameters, we also provide an efficient algorithm to generate an initial effective $\mathfrak{O}$-orientation, something that can always be done in polynomial time (using the maximal-order-to-supersingular-elliptic-curve algorithm from [31]) but might be very costly in practice.

Our new group action still requires a precomputation of complexity $L_\Delta(1/2)$: Here the main algorithmic task is to compute a *lattice of relations* for the class group, which can be used later to obtain a "short representative" of any given class in $\mathrm{Cl}(\mathfrak{O})$. Computing relations in the class group amounts to solving discrete logarithms in a subgroup of some finite field (unrelated to $\mathbb{F}_p$), whose order we can somewhat control by choosing the conductor.

Despite the fact that our choice of conductor is very constrained by the requirements on the generator α (see Section 5.1), we show that we have a search space large enough to obtain a fairly smooth class number. Thus, we were able to instantiate the SCALLOP group action for security levels that remain entirely out of reach for the CSI-FiSh approach, using only modest computational resources. Concretely, we give parameters for our group action with security comparable to CSIDH-512 and CSIDH-1024. This leads to an isogeny-based Fiat–Shamir signature analogous to CSI-FiSh for parameters twice as large as CSI-FiSh.

1.2 Technical overview

We give below a list of tasks and constraints required to create a setup analogous to CSI-FiSh. Then, we briefly explain how our new group action is evaluated and how it compares to CSI-FiSh.

We distinguish two phases in setting-up an isogeny-based group action: an offline and an online one. The offline phase is the main novelty introduced in CSI-FiSh compared to CSIDH [13]. It is performed just once at parameter generation. We do not need it to be efficient, but we want it to be feasible. This precomputation is crucial to the efficiency of the online phase, which is executed at every group action evaluation (hence dozens of times for each signature) and needs to be as fast as possible.

In the following, let $\mathfrak{O}$ be an imaginary quadratic order.

Evaluating isogeny group actions. Abstractly, a group action is defined by a group G, a set X, and a map $\star : G \times X \to X$ satisfying some set of axioms.

Algorithmically, we ask that elements of G and X have a representation, and that for any $g \in G$ and $x \in X$ it is feasible to compute $g \star x$. These, and other requirements, have been formalized under the names of Hard Homogenous Space (HHS) [21] or Effective Group Action (EGA) [1].

In the specific case of isogeny actions, the set X is a set of elliptic curves, which can be represented by an appropriate invariant, e.g. the j-invariant. The group $G = \mathrm{Cl}(\mathfrak{O})$ tends to be cyclic, or nearly cyclic, thus its elements could be uniquely represented as powers $\mathfrak{a}^e$ of some generator $\mathfrak{a}$. However it is not true in general that $\mathfrak{a}^e \star E$ can be efficiently evaluated for every exponent e and every curve E. Instead, there exist a list of ideals $\mathfrak{l}_1, \ldots, \mathfrak{l}_n$ of small norm, spanning $\mathrm{Cl}(\mathfrak{O})$ and such that the actions $\mathfrak{l}_i \star E$ can be efficiently evaluated for every $\mathfrak{l}_i$ and every E. Then, the action of any ideal of the form $\mathfrak{b} = \prod_{i=1}^n \mathfrak{l}_i^{e_i}$ can be efficiently evaluated as soon as the *exponent vector* $(e_1, \ldots, e_n) \in \mathbb{Z}^n$ has small norm. This setup is called a Restricted EGA (REGA) in [1].

To go from a REGA to an EGA, we need a way to rewrite any ideal class $\mathfrak{a}^e$ as a product $\mathfrak{a}^e = \prod_{i=1}^n \mathfrak{l}_i^{e_i}$ with small exponents. The main advance in CSI-FiSh was the computation of the *lattice of relations* of the ideals $\mathfrak{l}_1, \ldots, \mathfrak{l}_n$ in CSIDH-512, i.e. the lattice $\mathcal{L}$ spanned by the vectors $(e_1, \ldots, e_n)$ such that $\prod \mathfrak{l}_i^{e_i}$ is principal. If the $\mathfrak{l}_i$ span $\mathrm{Cl}(\mathfrak{O})$, then $\mathbb{Z}^n/\mathcal{L}$ is isomorphic to $\mathrm{Cl}(\mathfrak{O})$. Then, assuming $\mathfrak{a} = \mathfrak{l}_1$, finding a decomposition of $\mathfrak{a}^e$ with short exponents amounts to solving a Closest Vector Problem (CVP) in the lattice of relations for the vector $(e, 0, \ldots, 0)$.

Our aim is to replicate this strategy for the relation lattices associated to the class groups we are interested in.

The offline phase. The goal of this phase is to precompute the relation lattice of the class group $\mathrm{Cl}(\mathfrak{O})$, and produce a reduced basis of it. The main steps are:

1. Compute the class number and the structure of the class group.
2. Generate the lattice of relations $\mathcal{L}$.
3. Compute a reduced basis of $\mathcal{L}$ suitable for solving approximate-CVP.

In CSI-FiSh, the first item is obtained as a byproduct of the second, which is performed using index calculus, for an asymptotic cost of $L_\Delta(1/2)$. The last step is a standard lattice-basis reduction (typically done using BKZ); although, depending on the approximation factor, this step may even have exponential complexity, it is the fastest one in practice.

In this work we change the way the first two steps are performed. First, we choose $\mathfrak{O}$ so that the class group structure comes for free: We select a quadratic order $\mathfrak{O} = \mathbb{Z} + f\mathfrak{O}_0$ of large conductor f inside a maximal quadratic order $\mathfrak{O}_0$ of small discriminant d_0. Computing the class group structure, then, amounts to factoring f, which we choose to be a prime.

Secondly, by choosing $\mathfrak{O}_0$ and f carefully, not only can we compute the group structure, but we can even control it to some extent. In particular, we search for the prime f such that the class number of $\mathfrak{O}$, given by $f - \left(\frac{d_0}{f}\right)$, is somewhat smooth, so that computing discrete logarithms in $\mathrm{Cl}(\mathfrak{O})$ becomes feasible.

Then, instead of using index calculus, we directly obtain the lattice of relations by computing the discrete logarithm relationships between the generators $\mathfrak{l}_1, \ldots, \mathfrak{l}_n$. Asymptotically, an $L_f(1/2)$-long search for f is expected to yield an $L_f(1/2)$-smooth class number: At this level of detail in the analysis, no obvious improvement over index calculus stands out, however the constants hidden in the exponents turn out to be much more favorable to our setup, as our experiments confirm.

The final step, BKZ reduction, is unchanged.

In the online phase we evaluate the group action. The inputs are an oriented curve E and an integer e, the output is the oriented curve $\mathfrak{a}^e \star E$, where $\mathfrak{a}$ is some fixed generator (e.g. $\mathfrak{a} = \mathfrak{l}_1$). This phase consists of two steps:

1. Solving approximate-CVP to find a decomposition $\mathfrak{a}^e = \prod \mathfrak{l}_i^{e_i}$ with small exponents.
2. Using isogeny computations to evaluate $\left(\prod \mathfrak{l}_i^{e_i}\right) \star E$.

In SCALLOP the first step is identical to CSI-FiSh: We use Babai's nearest plane algorithm [4] to find a vector close to $(e, 0, \ldots, 0)$. The cost of this step is negligible, however the quality of the output depends on the quality of the basis computed in the offline phase, and has a big impact on the cost of the next step. In practice, the dimension of the lattices we consider is small enough that we can compute a nearly optimal basis, thus the approximation factor for CVP will be rather small. However, from an asymptotic point of view, there is a trade-off between the time spent reducing the lattice in the offline phase, and the approximation factor achieved in the online phase. The break-even point happens at $L(1/2)$, exactly like in CSI-FiSh.

The isogeny computation step is where we deviate most from CSI-FiSh. Indeed in CSI-FiSh there is an implicit orientation by the order $\mathfrak{O} = \mathbb{Z}[\sqrt{-p}]$, which is easily computed via Frobenius endomorphisms. In contrast, in SCALLOP we need an explicit representation of the orientation, that we transport along the group action. It is thus not surprising that, for the same parameter sizes, our algorithms are significantly slower than CSI-FiSh. Nonetheless we show there are choices of orientations for which it is at least feasible to run them.

Concretely, we choose a quadratic order $\mathfrak{O}$ that contains a generator α of smooth norm of size roughly equal to $\mathrm{disc}(\mathfrak{O})$ (essentially, the smallest size we could hope for). The orientation is then represented by an endomorphism ω corresponding to the principal ideal $\mathfrak{O}\alpha$, encoded as the composition of two isogenies of degree roughly equal to $\sqrt{\mathrm{disc}(\mathfrak{O})}$. The endomorphism ω plays here the same role as the Frobenius endomorphism in CSI-FiSh: An ideal $\mathfrak{l}_i$ acts through an isogeny of degree ℓ_i whose kernel is stabilized by ω, to compute $\mathfrak{l}_i \star E$ it is thus sufficient to evaluate ω on $E[\ell_i]$ and determine its eigenspaces.

In Section 5.1, we justify the concrete choices for $\mathfrak{O}$ in more detail and we present all required precomputations. The full description of the algorithm for the online phase is given in Section 5.2.

Organisation of the paper. The rest of this paper is organized as follows. Section 2 introduces the necessary mathematical background. In Section 3, we introduce our generic framework for effective orientation and group action computation. Then, we introduce the security notions related to group actions in Section 4. In Section 5, we explain in detail how the SCALLOP group action works. In Section 6, we discuss the concrete instantiation of the scheme. Finally, we analyze one particular angle of attack against the scheme in Section 7.

2 Preliminaries

2.1 Elliptic curves and isogenies

Elliptic curves. In this work we consider elliptic curves defined over a finite field $\mathbb{F}_{p^2}$, which can be represented, for example, by a Weierstrass equation

$$E \ : \ y^2 = x^3 + ax + b, \ \ a, b \in \mathbb{F}_{p^2}.$$

For a field extension k of $\mathbb{F}_{p^2}$, we write $E(k)$ for the group of k-rational points of E. We denote by $[n]P$ the nth scalar multiple of a point P, and by $E[n]$ the n-torsion subgroup of $E(\overline{\mathbb{F}}_{p^2})$, so $E[n] \simeq (\mathbb{Z}/n\mathbb{Z})^2$ as soon as $p \nmid n$.

Isogenies. An *isogeny* $\varphi : E_1 \to E_2$ is a non-constant morphism sending the identity of E_1 to that of E_2. The degree of an isogeny is its degree as a rational map (see [50]). An isogeny of degree d, or d-isogeny, is necessarily *separable* when $p \nmid d$, which implies $d = \#\ker\varphi$. An isogeny is said to be *cyclic* when its kernel is a cyclic group. For any $\varphi : E_1 \to E_2$, there exists a unique dual isogeny $\hat{\varphi} : E_2 \to E_1$, satisfying $\varphi \circ \hat{\varphi} = [\deg(\varphi)]$.

Endomorphism ring. An isogeny from a curve E to itself, or the constant zero map, is an *endomorphism*. The set $\mathrm{End}(E)$ of all endomorphisms of E forms a ring under addition and composition. For elliptic curves defined over a finite field, $\mathrm{End}(E)$ is isomorphic either to an order of a quadratic imaginary field or a maximal order in a quaternion algebra. In the first case, the curve is said to be *ordinary* and otherwise *supersingular*. We focus on the supersingular case, here, and we write $\mathcal{S}(p)$ for the set of isomorphism classes of supersingular curves defined over a field of characteristic p. It is a finite set containing roughly $p/12$ classes, and each class admits a representative over $\mathbb{F}_{p^2}$. The Frobenius isogeny $\pi : (x, y) \to (x^p, y^p)$ is the only inseparable isogeny between supersingular curves and it has degree p. We write $\pi : E \to E^p$. For any supersingular curve E we have $\mathrm{End}(E) \cong \mathrm{End}(E^p)$, but $E \cong E^p$ if and only if E has an isomorphic model over $\mathbb{F}_p$.

A concrete example: $j = 1728$. Let $p \equiv 3 \pmod 4$, and let $\mathcal{E}_0/\mathbb{F}_{p^2}$ be the curve $y^2 = x^3 + x$ of j-invariant 1728. Its endomorphism ring is isomorphic to the maximal order $\mathcal{O}_0 = \langle 1, i, \frac{i+j}{2}, \frac{1+k}{2}\rangle$ with $i^2 = -1$, $j^2 = -p$ and $k = ij$. Moreover, we have explicit endomorphisms π and ι such that $\mathrm{End}(E_0) = \langle 1, \iota, \frac{\iota+\pi}{2}, \frac{1+\iota\pi}{2}\rangle$, where π is the Frobenius isogeny and ι is the map $(x, y) \mapsto (-x, \sqrt{-1}y)$.

2.2 Representing and evaluating isogenies

Vélu's formulas [52] let us compute any separable isogeny φ of degree D, given $\ker\varphi$. They take $\widetilde{O}(\sqrt{D})$ operations over the field of definition of $\ker\varphi$ [6]. An isogeny of degree D can be decomposed into a sequence of isogenies of degrees the prime factors of D, thus the efficiency of any isogeny computation mainly depends on the largest prime factor $\ell|D$, and the size of the field extension containing $E[\ell]\cap\ker\varphi$. Hence, we will focus on isogenies of smooth degree where the related torsion groups $E[\ell]$ are defined over $\mathbb{F}_{p^2}$.

In practice, we encode cyclic D-isogenies as tuples (E, P), where P is a generator of $\ker\phi$. We call this a *kernel representation*. It can be compressed to only $O(\log(p) + \log(D))$ bits using techniques similar to SIDH key compression [3,20,55,45] (even when P is defined over a large field extension of $\mathbb{F}_{p^2}$). Such compression becomes relevant when large degree isogenies are exchanged as part of a key agreement message or cryptographic signature.

Pullback and push-forward. Let us take two coprime integers A, B. Any isogeny of degree AB can be factored in two ways as $\varphi_A\circ\varphi_B$ or $\psi_B\circ\psi_A$, where φ_A, ψ_A (resp. φ_B, ψ_B) have degree A (resp. B). This creates a commutative diagram, where $\ker\varphi_A = \varphi_B(\ker\psi_A)$ and $\ker\psi_B = \psi_A(\ker\varphi_B)$. Given ψ_A and φ_B we define φ_A (resp. ψ_B) as the *pushforward* of ψ_A through φ_B (resp. φ_B through ψ_A), which we denote by $\varphi_A = [\varphi_B]_*\psi_A$ (resp. $\psi_B = [\psi_A]_*\varphi_B$). There is also a dual notion of *pullback*, denoted by $[\cdot]^*\cdot$, so that $\psi_A = [\varphi_B]^*\varphi_A$ and $\varphi_B = [\psi_A]^*\psi_B$.

2.3 Orientation of supersingular curves and ideal group action

For the rest of this article, we fix a quadratic imaginary field K and a quadratic order $\mathfrak{O}$ of discriminant $D < 0$ in K. We will consider primitive $\mathfrak{O}$-orientations of supersingular curves. The notion of orientation in Definition 1 below corresponds to that of *primitive orientation with a p-orientation* in [19], and it is equivalent under the Deuring correspondence to *optimal embeddings* of quadratic orders inside maximal orders of $B_{p,\infty}$. The same notion is referred to as normalized optimal embeddings in [5].

Definition 1. *For any elliptic curve E, a K-orientation is a ring homomorphism $\iota : K \hookrightarrow \mathrm{End}(E)\otimes\mathbb{Q}$. A K-orientation induces an $\mathfrak{O}$-orientation if $\iota(\mathfrak{O}) = \mathrm{End}(E)\cap\iota(K)$. In that case, the pair (E, ι) is called a $\mathfrak{O}$-oriented curve and E is an $\mathfrak{O}$-orientable curve.*

In what follows, we consider the elements of $\mathcal{S}(p)/\pi$ rather than $\mathcal{S}(p)$ because the Frobenius π creates two orientations (one in E and one in $E^{(p)}$) from each optimal embedding of $\mathfrak{O}$ in a maximal quaternion order of $B_{p,\infty}$. Note that this is not the convention taken in [46,54], where orientations are not considered up to Galois conjugacy.

Definition 2. *$\mathcal{S}_{\mathfrak{O}}(p)$ is the set of $\mathfrak{O}$-oriented curves (E, ι) up to isomorphisms and Galois conjugacy.*

The following proposition follows from the results proven by Onuki [46, Proposition 3.2, Proposition 3.3, Theorem 3.4] and gives a way to compute $\#\mathcal{S}_{\mathfrak{O}}(p)$.

Proposition 3. *The set $\mathcal{S}_{\mathfrak{O}}(p)$ is not empty if and only if p does not split in K and does not divide the conductor of $\mathfrak{O}$. When these conditions are satisfied, and p is not ramified in K, we have $\#\mathcal{S}_{\mathfrak{O}}(p) = h(\mathfrak{O})$.*

When p is ramified in K, the situation is a bit more complicated but it can be shown [2] that

$$\#\mathcal{S}_{\mathfrak{O}}(p) \in \left\{ \frac{1}{2}h(\mathfrak{O}), h(\mathfrak{O}) \right\}.$$

When $\mathcal{S}_{\mathfrak{O}}(p)$ is not empty, the set of invertible $\mathfrak{O}$-ideals acts on $\mathfrak{O}$-orientations via an operation that we write $\mathfrak{a} \star (E, \iota) = (E_{\mathfrak{a}}, \iota_{\mathfrak{a}})$. Principal ideals act trivially, thus the operation $\star$ defines a group action of $\mathrm{Cl}(\mathfrak{O})$ on $\mathcal{S}_{\mathfrak{O}}(p)$, which we also denote by $\star$. Onuki proved that this group action is free and transitive.

Concretely, this action is computed using isogenies. For an ideal $\mathfrak{a}$ in $\mathfrak{O}$ and an $\mathfrak{O}$-orientation (E, ι_E), we define $E[\mathfrak{a}] = \bigcap_{\alpha \in \mathfrak{a}} \ker \iota_E(\alpha)$ and write $\varphi_{\mathfrak{a}}^E$ for the isogeny of kernel $E[\mathfrak{a}]$. We have

$$\varphi_{\mathfrak{a}}^E : E \to E_{\mathfrak{a}} = E/E[\mathfrak{a}] \quad \text{and} \quad \iota_{\mathfrak{a}}^E(x) = \frac{1}{n(\mathfrak{a})} \varphi_{\mathfrak{a}}^E \circ \iota(x) \circ \hat{\varphi}_{\mathfrak{a}}^E. \tag{1}$$

When $\mathfrak{a}$ does not factor as $n\mathfrak{b}$ for some integer $n > 1$, we say that $\mathfrak{a}$ is *primitive*. In that case, the corresponding isogeny $\varphi_{\mathfrak{a}}^E$ is cyclic.

It will be useful for us to consider a generator α of $\mathfrak{O}$ (an element such that $\mathfrak{O} = \mathbb{Z}[\alpha]$). In that case, every ideal $\mathfrak{a}$ can be written as $\langle x + \alpha y, n(\mathfrak{a}) \rangle$ for some $x, y \in \mathbb{Z}$. Note that this choice of generator is not unique: if α is a generator, any $\alpha + k$ will also be generators with $k \in \mathbb{Z}$.

Although an orientation may exist it is not always clear how to represent it and compute with it. Informally, an *effective orientation* is one that comes with efficient representations and algorithms. We will give a more formal, and slightly more specific definition in Section 3.2.

3 The generic group action

This section introduces our general framework for evaluating group actions of oriented curves. The algorithm we outline below is not designed to be particularly efficient. Later, in Section 5.2, we will describe in detail a version and parameter choices that make it somewhat practical.

The key to our technique is having a generator of smooth norm for the quadratic order. To simplify the exposition, we restrict to quadratic orders $\mathfrak{O}$ with a generator α of norm $L_1^2 L_2$, where L_1 and L_2 are two smooth coprime integers and the principal ideal $\mathfrak{O}\alpha$ is equal to $\mathfrak{L}_1^2 \mathfrak{L}_2$ for some primitive ideals $\mathfrak{L}_1, \mathfrak{L}_2$. We will further refine these constraints in Section 5.1 for an efficient instantiation.

We now present a few generic properties, then, in Section 3.2, we describe how the orientation by such an order can be made effective.

3.1 Factorization of ideals and decomposition of isogenies

We recall from Eq. (1) that if (E, ι_E) is an oriented curve and $\mathfrak{a}$ is an ideal, the action $\mathfrak{a} \star (E, \iota_E)$ is computed via an isogeny denoted by $\varphi_{\mathfrak{a}}^E$.

Proposition 4. *If $\mathfrak{a}$ can be factored as $\mathfrak{a}_1\mathfrak{a}_2$, then the isogeny $\varphi_{\mathfrak{a}}^E$ can be decomposed as $\varphi_{\mathfrak{a}_2}^{E_{\mathfrak{a}_1}} \circ \varphi_{\mathfrak{a}_1}^E$. Moreover, if $\mathfrak{a}_1$ and $\mathfrak{a}_2$ have coprime norms, then $\varphi_{\mathfrak{a}_2}^{E_{\mathfrak{a}_1}} = [\varphi_{\mathfrak{a}_1}^E]_* \varphi_{\mathfrak{a}_2}^E$.*

Proof. The fact that we can factor $\varphi_{\mathfrak{a}}^E$ is standard and the formula to compute $\varphi_{\mathfrak{a}_2}^{E_{\mathfrak{a}_1}}$ follows from Lemma 5 below. □

Lemma 5. *Let $\mathfrak{a}, \mathfrak{b}$ be two ideals such that $E[\mathfrak{a}] \cap E[\mathfrak{b}] = \{0\}$. Let $\varphi_{\mathfrak{a}}^E : E \to E_{\mathfrak{a}} := E/E[\mathfrak{a}]$ be the isogeny corresponding to the action of $\mathfrak{a}$ on (E, ι_E). Then $E_{\mathfrak{a}}[\mathfrak{b}] = \varphi_{\mathfrak{a}}^E(E[\mathfrak{b}])$.*

Proof. Firstly, let us suppose that $a = n(\mathfrak{a})$ and $b = n(\mathfrak{b})$ are coprime. Then the lemma follows from the usual commutative diagram obtained by decomposing the isogeny $\varphi_{\mathfrak{a}\mathfrak{b}}^E$ as $\varphi_{\mathfrak{b}}^{E_{\mathfrak{a}}} \circ \varphi_{\mathfrak{a}}^E$ with $E_{\mathfrak{a}}[\mathfrak{b}] = \ker \varphi_{\mathfrak{b}}^{E_{\mathfrak{a}}} = \varphi_{\mathfrak{a}}^E(E[\mathfrak{b}])$.

Secondly, let us suppose that $a = b$. Then since $E[\mathfrak{a}] \cap E[\mathfrak{b}] = \{0\}$, we have $\mathfrak{b} = \overline{\mathfrak{a}}$ and $\mathfrak{b} \star \mathfrak{a} \star (E, \iota_E) = (E, \iota_E)$. It follows that $E_{\mathfrak{a}}[\mathfrak{b}] = E_{\mathfrak{a}}[\overline{\mathfrak{a}}] = \ker \hat{\varphi}_{\mathfrak{a}}^E = \varphi_{\mathfrak{a}}^E(E[a]) = \varphi_{\mathfrak{a}}^E(E[\mathfrak{a}] \oplus E[\mathfrak{b}]) = \varphi_{\mathfrak{a}}^E(E[\mathfrak{b}])$.

Lastly, suppose generally that $\gcd(a, b) = c$, writing $a = ca'$, $b = cb'$, $\mathfrak{a} = \mathfrak{c}\mathfrak{a}'$ and $\mathfrak{b} = \overline{\mathfrak{c}}\mathfrak{b}'$. Then $E_{\mathfrak{a}}[\mathfrak{b}] = E_{\mathfrak{a}}[\overline{\mathfrak{c}}] \oplus E_{\mathfrak{a}}[\mathfrak{b}']$. Combining the first case and the second one, we have $E_{\mathfrak{a}}[\overline{\mathfrak{c}}] = \varphi_{\mathfrak{c}}^{E_{\mathfrak{a}'}}(E_{\mathfrak{a}'}[\overline{\mathfrak{c}}]) = \varphi_{\mathfrak{c}}^{E_{\mathfrak{a}'}} \circ \varphi_{\mathfrak{a}'}^E(E[\overline{\mathfrak{c}}]) = \varphi_{\mathfrak{a}}^E(E[\overline{\mathfrak{c}}])$ and $E_{\mathfrak{a}}[\mathfrak{b}'] = \varphi_{\mathfrak{c}}^{E_{\mathfrak{a}'}}(E_{\mathfrak{a}'}[\mathfrak{b}']) = \varphi_{\mathfrak{c}}^{E_{\mathfrak{a}'}} \circ \varphi_{\mathfrak{a}'}^E(E[\mathfrak{b}']) = \varphi_{\mathfrak{a}}^E(E[\mathfrak{b}'])$. Hence $E_{\mathfrak{a}}[\mathfrak{b}] = \varphi_{\mathfrak{a}}^E(E[\mathfrak{b}])$. □

When using Lemma 5, we will in general specify the tuple $(E, \mathfrak{a}, \mathfrak{b})$ at hand.

3.2 Effective orientation

Let us take an $\mathfrak{O}$-orientation (E, ι_E). Through ι_E, we obtain an endomorphism $\omega_E \in \mathrm{End}(E)$ as $\iota_E(\alpha)$. This endomorphism ω_E has degree $L_1^2 L_2$ and it can be decomposed as $\omega_E = \hat{\varphi}_{\mathfrak{L}_1^{-1}}^E \circ \varphi_{\mathfrak{L}_1\mathfrak{L}_2}^E$, as Proposition 4 shows. Thus, we obtain a representation of ω_E from the kernel representations of the two isogenies $\hat{\varphi}_{\mathfrak{L}_1^{-1}}^E$ and $\varphi_{\mathfrak{L}_1\mathfrak{L}_2}^E$. This idea of decomposing an endomorphism into a cycle of two isogenies is now quite standard in isogeny-based cryptography (see for instance [48,27]).

Formally, we have the following definition.

Definition 6. *Let $(E, \iota_E) \in \mathcal{S}_{\mathfrak{O}}(p)$ where $\mathfrak{O} = \mathbb{Z}[\alpha]$ with $\alpha = \mathfrak{L}_1^2\mathfrak{L}_2$. An* effective orientation *for (E, ι_E) is a tuple $s_E = (E, P_E, Q_E)$ where (E, P_E) and (E, Q_E) are the respective kernel representations of the isogenies $\varphi_{\mathfrak{L}_1\mathfrak{L}_2}^E$ and $\varphi_{\mathfrak{L}_1}^E$ (of respective degree L_1L_2 and L_1).*

Remark 7. When it comes to using an effective orientation as public key, it is important to represent it in a canonical way. For example, when performing a key exchange with SCALLOP, the shared key (which is an oriented curve), must be canonically represented so that both parties can get the same shared key. Given $s_E = (E, P_E, Q_E)$, one computes canonical generators P'_E and Q'_E of the groups $\langle P_E \rangle$ and $\langle Q_E \rangle$ respectively. The effective representation $s'_E = (E, P'_E, Q'_E)$ is then refered to as the canonical effective representation for (E, ι_E).

Since L_1 and L_2 are coprime, $P_E = R_E + S_E$ where R_E and S_E are points of order L_1 and L_2 respectively. Given P_E, one recovers $R_E = [\lambda_2 L_2]P_E$ and $S_E = [\lambda_1 L_1]P_E$, where λ_1 is the inverse of L_1 mod L_2 and λ_2 is the inverse of L_2 mod L_1. Conversely, given R_E and S_E, one recovers $P_E = R_E + S_E$. In some cases, such as the statement and proof of Proposition 9, we may directly assume ω_E is represented as (R_E, S_E, Q_E), for simplicity.

3.3 The group action computation from the effective orientation

Let $\mathfrak{a}$ be an ideal of $\mathfrak{O}$, our goal now is to understand how to compute an effective orientation $\omega_{E_\mathfrak{a}}$ for $\mathfrak{a} \star (E, \iota_E)$ from the effective orientation ω_E.

By Proposition 4, we know that we can focus on the case where $\mathfrak{a} = \mathfrak{l}$ is a prime ideal. If we know how to compute $\varphi_\mathfrak{l}^E$ and the effective orientation $\omega_{E_\mathfrak{l}}$ for $(E_\mathfrak{l}, \iota_{E_\mathfrak{l}}) = \mathfrak{l} \star (E, \iota_E)$, from $\mathfrak{l}$ and ω_E, then we can recursively compute the action of any ideal $\mathfrak{a}$ from its factorization as a product of prime ideals. So we need to focus on two operations: computing $\varphi_\mathfrak{l}^E$ and computing $\omega_{E_\mathfrak{l}}$.

Computation of the group action isogeny. The computation of $\varphi_\mathfrak{l}^E$ can be done from $\ker \varphi_\mathfrak{l}^E = E[\mathfrak{l}]$ using Vélu's formulas [52]. Thus, the main operation is the computation of $E[\mathfrak{l}]$ from ω_E. Proposition 8 provides this operation.

Proposition 8. *When ℓ is split in $\mathfrak{O} = \mathbb{Z}[\alpha]$, and $\mathfrak{l}$ is a prime ideal above ℓ, there exists $\lambda \in \mathbb{Z}$ such that $\mathfrak{l} = \langle \alpha - \lambda, \ell \rangle$. Then, $\ker \varphi_\mathfrak{l}^E = E[\mathfrak{l}] = E[\ell] \cap \ker \rho_E$ where $\rho_E = \omega_E - [\lambda]_E$.*

Proof. It suffices to see that $n(\alpha - \lambda) = \lambda^2 - \lambda \mathrm{tr}(\alpha) + n(\alpha)$ has two solutions modulo ℓ if and only if $\mathrm{disc}\mathfrak{O} = \mathrm{tr}(\alpha)^2 - 4n(\alpha)$ is a non-zero square modulo ℓ which is exactly the case where ℓ splits in $\mathfrak{O}$. The ideal $\langle \alpha - \lambda, \ell \rangle$ has norm ℓ because $\alpha - \lambda \notin \ell\mathfrak{O}$ (because ℓ is split in $\mathfrak{O}$). Then the result follows from the definition of $\varphi_\mathfrak{l}^E$. □

The computation of a generator of $\ker \varphi_\mathfrak{l}^E$ from Proposition 8 is quite standard in the literature on isogeny-based cryptography. It suffices, for instance, to evaluate $\omega_E - [n(\alpha)/\lambda]$ (or $\omega_E - \mathrm{tr}(\alpha)$ if $\lambda = 0$) on a basis P, Q of $E[\ell]$, then at least one of the two images will generate $E[\mathfrak{l}]$. From this, we derive the kernel representation of $\varphi_\mathfrak{l}^E$.

Computation of the new effective orientation. Computing $\omega_{E_\mathfrak{l}}$ is less straightforward. There are basically two cases depending on whether ℓ is coprime with $n(\alpha) = \deg \omega_E$ or not. The first case is by far the simplest: When ℓ and $n(\alpha)$ are coprime, applying Proposition 4 to $\omega_E = \hat{\varphi}^E_{\mathfrak{L}_1^{-1}} \circ \varphi^E_{\mathfrak{L}_1\mathfrak{L}_2}$ shows that $\omega_{E_\mathfrak{l}} = [\varphi^E_\mathfrak{l}]_* \omega_E$. Thus, it suffices to push the generators of $\hat{\varphi}^E_{\mathfrak{L}_1^{-1}}$ and $\varphi^E_{\mathfrak{L}_1\mathfrak{L}_2}$ through $\varphi^E_\mathfrak{l}$ to get a kernel representation for $\omega_{E_\mathfrak{l}}$.

The story is more complicated when ℓ and $n(\alpha)$ are not coprime because the pushforward of ω_E is not well-defined in this case. Let us treat the simplified case where $L_1 = \ell$ (and so $n(\alpha) = \ell^2 L_2$ for some L_2 coprime with ℓ), as the generic case can be handled with similar ideas. The full algorithm to handle the generic case is given in Section 5.2.

When $n(\alpha) = \ell^2 L_2$, there are two possibilities: either $\mathfrak{L}_1 = \mathfrak{l}$ or $\mathfrak{L}_1 = \mathfrak{l}^{-1}$ as there are no further primitive ideals of norm dividing ℓ. If we have a method to solve the former, we can derive a method to solve the latter by considering the dual of the endomorphism ω_E. Thus, we focus on $\mathfrak{L}_1 = \mathfrak{l}$.

Proposition 9. *Let α be a generator of $\mathfrak{O}$ of norm $\ell^2 L_2$ with $\mathfrak{O}\alpha = \mathfrak{l}^2\mathfrak{L}_2$ as above. Then $\omega_E = \iota(\alpha)$ can be decomposed as $\hat{\varphi}^E_{\bar{\mathfrak{l}}} \circ \varphi^{E_\mathfrak{l}}_{\mathfrak{L}_2} \circ \varphi^E_\mathfrak{l}$. Suppose that ω_E is represented as (R_E, S_E, Q_E) where $E[\mathfrak{l}] = \langle R_E \rangle$, $E[\mathfrak{L}_2] = \langle S_E \rangle$ and $E[\bar{\mathfrak{l}}] = \langle Q_E \rangle$. The effective orientation of the curve $E_\mathfrak{l}$ is $(R_{E_\mathfrak{l}}, S_{E_\mathfrak{l}}, Q_{E_\mathfrak{l}})$ where:*

$$\begin{aligned} Q_{E_\mathfrak{l}} &= \varphi^E_\mathfrak{l}(Q_E) \\ R_{E_\mathfrak{l}} &= \widehat{\varphi^{E_\mathfrak{l}}_{\mathfrak{L}_2}} \circ \varphi^E_{\bar{\mathfrak{l}}}(R_E) \\ S_{E_\mathfrak{l}} &= \varphi^E_\mathfrak{l}(S_E). \end{aligned}$$

Proof. By the definitions of the group action and of the effective orientation, $\omega_E = \iota(\alpha)$ implies $\omega_{E_\mathfrak{l}} = \iota_\mathfrak{l}(\alpha)$. This is why we obtain the two decompositions $\hat{\varphi}^E_{\bar{\mathfrak{l}}} \circ \varphi^{E_\mathfrak{l}}_{\mathfrak{L}_2} \circ \varphi^E_\mathfrak{l}$ for ω_E and $\hat{\varphi}^{E_\mathfrak{l}}_{\bar{\mathfrak{l}}} \circ \varphi^{E_{\mathfrak{l}^2}}_{\mathfrak{L}_2} \circ \varphi^{E_\mathfrak{l}}_\mathfrak{l}$ for $\omega_{E_\mathfrak{l}}$ from the factorization $\mathfrak{O}\alpha = \mathfrak{l}^2\mathfrak{L}_2$. The rest of the proposition follows by applying Lemma 5 to $(E, \mathfrak{l}, \bar{\mathfrak{l}})$, $(E, \overline{\mathfrak{L}_2\mathfrak{l}}, \mathfrak{l})$, and $(E, \mathfrak{l}, \mathfrak{L}_2)$ respectively. □

Note that Proposition 9 remains valid when we replace the ideal $\mathfrak{l}$ by any ideal of smooth norm dividing $\alpha\mathfrak{O}$. This will be the case in Section 5 where we evaluate the action of a product of prime ideals $\mathfrak{l}_i$ where some $\mathfrak{l}_i^2$ divide $\alpha\mathfrak{O}$ and others do not.

In Section 5, we introduce a concrete instantiation of the general principle described above. There, we provide a detailed and efficient version of the algorithms outlined in this section.

Comparison with CSIDH. In CSIDH [13], the effective orientation is obtained through the Frobenius endomorphism, which has norm p and is thus coprime to the norm of all ideals we need to evaluate. Thus, we are in the easy case. Moreover, the situation of CSIDH is particularly simple because the kernel of $\varphi^E_\mathfrak{l}$ can be directly obtained as one of the two subgroups of order ℓ stable under Frobenius.

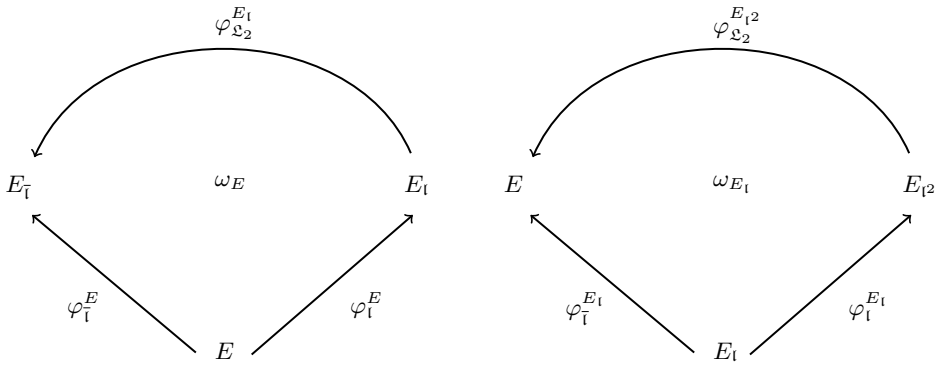

Fig. 1. A picture of the effective orientation computation from Proposition 9.

4 Security of a group action

In this section, we define the security notions associated to a cryptographic group action — a (very) hard homogenous space — and we review the best known attacks. A (free and transitive) group action $\star$ of a group G on a set X is a *hard homogenous space* if it can be computed efficiently and the following problems are hard.

Problem 10 (Vectorisation). Given $x, y \in X$, find $g \in G$ such that $y = g \star x$.

Problem 11 (Parallelisation). Given $x, g\star x, h\star x \in X$ (for undisclosed $g, h \in G$), find $(gh) \star x$.

It is a *very hard homogenous space* if the following problem is hard.

Problem 12 (Decisional Parallelisation). Given $x, y, u, v \in X$, decide whether there exists $g \in G$ such that $y = g \star x$ and $v = g \star u$.

The vectorisation and parallelisation problems, when instantiated with our group action of the class group of $\mathfrak{O}$ on $\mathcal{S}_\mathfrak{O}(p)$, are also known as the problems $\mathfrak{O}$-Vectorisation and $\mathfrak{O}$-DiffieHellman, studied in [54]. For simplicity, assume that the factorization of $\mathrm{disc}(\mathfrak{O})$ is known, and that it has $O(\log\log|\mathrm{disc}(\mathfrak{O})|)$ distinct prime factors[14], as will be the case of our construction.

The two problems $\mathfrak{O}$-Vectorisation and $\mathfrak{O}$-DiffieHellman are equivalent under quantum reductions (see [34,44] for reductions that are polynomial in the cost of evaluating the group action, or [54] for reductions that are polynomial in the instance lengths).

Furthermore, these problems are closely related to the endomorphism ring problem, a foundational problem of isogeny-based cryptography: given a supersingular curve E, compute a basis of the endomorphism ring $\mathrm{End}(E)$ (i.e., four

[14] Note that the average number of distinct prime factors of integers up to n is indeed $O(\log\log n)$.

endomorphisms of E that generate $\mathrm{End}(E)$ as a lattice). More precisely, the problem $\mathfrak{O}$-Vectorisation is equivalent to the following oriented version of the endomorphism ring problem (see [54, Figure 1]).

Problem 13 ($\mathfrak{O}$-EndRing). Given an effectively oriented curve $(E, \iota_E) \in \mathcal{S}_{\mathfrak{O}}(p)$, compute a basis of the endomorphism ring $\mathrm{End}(E)$.

Clearly, $\mathfrak{O}$-EndRing reduces to the standard endomorphism ring problem, but the converse is not known to be true. In fact, $\mathfrak{O}$-EndRing currently seems simpler than the endomorphism ring problem as long as $|\mathrm{disc}(\mathfrak{O})| < p^2$. Precisely,

- The endomorphism ring problem can be solved in time $(\log p)^{O(1)} p^{1/2}$ (see [30,32]), and
- The problem $\mathfrak{O}$-EndRing can be solved in time $l^{O(1)} |\mathrm{disc}(\mathfrak{O})|^{1/4}$ with l the length of the input (see [54, Proposition 3]).

Write $\mathfrak{O} = \mathbb{Z} + f\mathfrak{O}_0$ where f is the conductor of $\mathfrak{O}$ and $\mathfrak{O}_0$ is the maximal order. Better algorithms than the above are known when $\mathfrak{O}_0$ has small class group and f is powersmooth (see [54, Theorem 5]), or even smooth in certain situations (as discussed in [19], or more generally [54, Corollary 6]). We will protect against such attacks by choosing f a large prime. This is in fact one key difference between OSIDH [19] and our construction. In OSIDH [19], the setting is similar, but f is smooth (a power of two), and the f-torsion is defined over $\mathbb{F}_{p^2}$. For this not to be a vulnerability, OSIDH is forced to only reveal partial information on the orientations, which must be done carefully, lest the attacks of [23] apply. An unfortunate side effect is that, without the full orientation, OSIDH does not actually provide an effective group action.

In summary, the fastest known generic *classical* method to solve the vectorisation problem associated to the group action has complexity

$$\begin{aligned} &\min\left((\log p)^{O(1)} p^{1/2}, \log(p+d)^{O(1)} d^{1/4}\right) \\ &= \log(p+d)^{O(1)} \min\left(p^{1/2}, f^{1/2}\right), \end{aligned}$$

where $d = |\mathrm{disc}(\mathfrak{O})|$. A precise estimate of the $O(1)$ appearing in the complexity of [54, Proposition 3] would provide a more precise estimation of the cost of an attack.

Regarding quantum security, there is an asymptotically faster heuristic algorithm, which runs in subexponential time (see [54, Proposition 4]). It relies on Kuperberg's algorithm [38] for the Abelian hidden shift problem, and runs in time

$$\log(p)^{O(1)} L_{\mathrm{disc}(\mathfrak{O})}(1/2).$$

Note that in special cases the hidden shift problem can be solved in polynomial time as discussed in [17,36,14]. These include groups isomorphic to $(\mathbb{Z}/\ell\mathbb{Z})^k$ where ℓ is a small prime and groups of the form $(\mathbb{Z}/2\mathbb{Z})^k \times (\mathbb{Z}/q\mathbb{Z})^r$ where q is a small prime. In general class groups rarely have this structure and for the parameter sets proposed, we can easily see that these attacks do not apply.

Finally, let us discuss the hardness of the Decisional Parallelisation problem. Clearly, it is not harder than vectorisation, hence the algorithms discussed above apply. The only known method that may outperform them is an algorithm to distinguish the action of ideal classes *up to squares*. More precisely, to each odd prime divisor $m \mid \mathrm{disc}(\mathfrak{O})$ is associated a quadratic character, i.e., a group homomorphism

$$\chi_m : \mathrm{Cl}(\mathfrak{O}) \longrightarrow \{\pm 1\},$$

Given oriented curves (E, ι) and $(E^{\mathfrak{a}}, \iota^{\mathfrak{a}})$, the algorithm of [12] (a generalisation of [15]) allows one to evaluate $\chi_m([\mathfrak{a}])$ in time polynomial in m. In fact, the algorithm requires finding random points in $E[m]$, and solving a discrete logarithm in a group of order m. Hence the quantum complexity may be as low as polynomial in $\log(m)$ and k if the points of $E[m]$ are defined over $\mathbb{F}_{p^k}$. There may also be two additional computable characters if $\mathrm{disc}(\mathfrak{O})$ is even. Clearly, if $[\mathfrak{a}] \in \mathrm{Cl}(\mathfrak{O})^2$ is a square, then $\chi_m([\mathfrak{a}]) = 1$, so one can prevent this attack by using $\mathrm{Cl}(\mathfrak{O})^2$ instead of $\mathrm{Cl}(\mathfrak{O})$. Another way to prevent this attack is to ensure that all prime factors of $\mathrm{disc}(\mathfrak{O})$ are large, and $E[m]$ lives in a large field extension, so no character can be computed efficiently.

5 SCALLOP: a secure and efficient group action

We finally propose an efficient instantiation of the effective group action outlined in Section 3. Our main algorithm is given in Section 5.2, but we need to motivate our parameter choices first. This is what we do in Section 5.1, where we also explain all the required precomputations.

5.1 Parameter choice and precomputation

The content of this section covers all the choices of parameters and precomputations required to make the SCALLOP group action computation secure and efficient. All the algorithms described here have to be run only once, at the moment of generating public parameters. We refer the reader to Section 1.2 for a list of all the requirements of that precomputation to obtain a construction similar to CSI-FiSh.

Choice of quadratic order. Our main motivation is to obtain a quadratic order $\mathfrak{O}$ of large discriminant, but with an easy to compute structure of the class group. In general, this is a very hard problem for classical computers, the best algorithm being index calculus, with a complexity of $L_{\mathrm{disc}(\mathfrak{O})}(1/2)$. But there are some special cases where the structure is easily determined, e.g. when

$$\mathfrak{O} = \mathbb{Z} + f\mathfrak{O}_0, \tag{2}$$

where $\mathfrak{O}_0$ is a quadratic maximal order of small discriminant and f is in $\mathbb{Z}$. In that case, we deduce directly the structure of $\mathrm{Cl}(\mathfrak{O})$ from that of $\mathrm{Cl}(\mathfrak{O}_0)$ and the factorization of f. In practice, we propose to take $\mathfrak{O}_0$ of class number one

(e.g. the Gauss integers) and f a prime number (also for security, as discussed in Section 4).

We give below a formula for the class number of such an order. The group structure, which turns out to be cyclic when $\mathfrak{O}_0$ has class number one, will be described in Appendix A.

Proposition 14. *Let f be a prime integer and let $\mathfrak{O}_0$ be a quadratic order of class number h_0, discriminant d_0 and let u_0 denote $|\mathfrak{O}_0^\times|/2$. The class number of $\mathbb{Z} + f\mathfrak{O}_0$ is equal to $\left(f - \left(\frac{d_0}{f}\right)\right)\frac{h_0}{u_0}$.*

Note that u_0 is one for all orders corresponding to curves with j-invariant different from 0 or 1728. From now on, we write d_0 for $\mathrm{disc}(\mathfrak{O}_0)$, and we assume the class number is one. It is not too difficult to generalize the algorithms below to larger class numbers, as long as d_0 is small.

Choice of conductor. We argued that we need a prime f for security, and to avoid factoring. Prime numbers also have the advantage of being abundant and easy to generate. Apart from this, our choice of f will be determined by efficiency constraints. In particular, to use the algorithm outlined in Section 3, we require that there exists a generator α with norm equal to $L_1^2 L_2$ to obtain effective $\mathfrak{O}$-orientations. Since the manipulation of this effective orientation requires computing L_1- and L_2-isogenies, we need L_1 and L_2 to be smooth. Moreover, we need L_2 to be small for the algorithm SetUpCurve described below.

Finally, there is a third requirement that we will motivate a bit later: that $f - (\frac{d_0}{f})$ is as smooth as possible. This last constraint impacts the efficiency of the offline phase of our scheme. As such, it is less important than the smoothness of $L_1 L_2$, which impacts the cost of the online phase. This is why our approach consists in finding a set of candidates for f that closely match the first two constraints, before sieving through the set to find the best candidate for the last requirement. In Section 6.1, we provide more details on how we select the parameters and we give some concrete examples of cryptographic size.

For a given $\mathfrak{O}$, finding a generator α of smooth norm is quite hard. Indeed, for a generic $\mathfrak{O}$, the size of the α of smallest smooth norm will be very large compared to f. This is why we choose the conductor f (and thus the order $\mathfrak{O}$) at the same time as the generator α. Our method allows us to find a conductor f and an α of smooth norm of optimal size (i.e, $n(\alpha) \approx f^2$). To do that, we first target a smooth norm $L_1^2 L_2$, and then we find a suitable conductor f.

Concretely, we fix a collection of principal ideals of small prime norm in $\mathfrak{O}_0$. Let us write α_0 for a generator of $\mathfrak{O}_0$ and $\mathfrak{l}_1, \ldots, \mathfrak{l}_m$ for the collection of principal ideals and $\ell_1, \ldots, \ell_m$ for the associated split primes. Because the ℓ_i are split, there are two principal ideals of norm ℓ_i in $\mathfrak{O}_0$: $\mathfrak{l}_i$ and its conjugate $\overline{\mathfrak{l}_i}$, which, by a slight abuse of notation, we write $\mathfrak{l}_i^{-1}$. We denote by L the product $\prod_{i=1}^m \ell_i$. For some $n_1 < n_2 \leq m$, we consider the products $\prod_{i=1}^{n_1} \mathfrak{l}_i^{b_i} \prod_{i=n_1+1}^{n_2} \mathfrak{l}_i^{c_i}$ where all $b_i \in \{-2, 2\}$ and $c_i \in \{-1, 1\}$, then we get 2^{n_2} principal ideals of norm $L_1^2 L_2$ with $L_1 = \prod_{i=1}^{n_1} \ell_i$ and $L_2 = \prod_{i=n_1+1}^{n_2} \ell_i$. It suffices to obtain one such ideal of the form $\langle L_1^2 L_2, \alpha \rangle$ where $\alpha = x + f\alpha_0$ for some prime number f to get that $\mathbb{Z} + f\mathfrak{O}_0 = \mathbb{Z}[\alpha]$

where α has norm $L_1^2 L_2$ as we desire. Each product has probability roughly $1/\log(L_1^2 L_2)$ to satisfy the desired property. This gives a set of size $2^m/\log(L)$ to sieve through in order to find the best candidate with respect to our third requirement (we have the estimate $m = O(\log(L_1^2 L_2)/\log\log(L_1^2 L_2))$, see for instance [35, Chapter 22]). Up to exchanging $\mathfrak{l}_i$ and $\mathfrak{l}_i^{-1}$, we can assume that all the b_i and c_i are positive and so we have $\mathfrak{O}_0\alpha = \prod_{i=1}^{n_1} \mathfrak{l}_i^2 \prod_{i=n_1+1}^{n_2} \mathfrak{l}_i$.

Remark 15. Note that for a fixed $\mathfrak{O}_0$ of discriminant d_0, the choice of class group determines $\left(\frac{d_0}{p}\right)$ to be 1 or -1. This is the only condition imposed on the prime characteristic p by the choice of class group. Thus, we will be able to choose p in a way that enables efficient computations after a suitable $\mathfrak{O}$ has been found.

Computing the relation lattice. Knowing the order of $\mathrm{Cl}(\mathfrak{O})$ is not enough for our application. Indeed, we want to be able to efficiently evaluate the action of any ideal class, which, by virtue of Proposition 4, calls for a way to compute for any class a representative that factors as a short product of ideals of small norm. For that, we follow the method introduced in [9].

The first step is to choose a set $\{\mathfrak{l}_1, \ldots, \mathfrak{l}_m, \ldots, \mathfrak{l}_n\}$ of $n = O(\log(f))$ ideals of small prime norm,[15] and to generate its *lattice of relations* $\mathcal{L}$, i.e. the lattice spanned by the vectors $(e_1, \ldots, e_n) \in \mathbb{Z}^n$ such that the ideal $\prod_{i=1}^n \mathfrak{l}_i^{e_i}$ is principal in $\mathfrak{O}$. [9] uses an index calculus method, with complexity $L_f(1/2)$, to compute a basis of $\mathcal{L}$. But another basis is simply given by the relations $\mathfrak{a}^{h(\mathfrak{O})} = 1$ and $\mathfrak{a}^{x_i} = \mathfrak{l}_i$, where $\mathfrak{a}$ is any generator of $\mathrm{Cl}(\mathfrak{O})$ and the x_i are the discrete logarithms to base $\mathfrak{a}$. If we force $\mathrm{Cl}(\mathfrak{O})$ to have smooth order, we can efficiently compute these discrete logarithms using the Pohlig–Hellman method.

This explains why we search for f such that $f - (\frac{d_0}{f})$ is as smooth as possible (recall Proposition 14). Unfortunately, we could not find a method to significantly bias $f - (\frac{d_0}{f})$ towards being smooth, thus our method still has subexponential complexity: Heuristically, if we sieve through $L_f(1/2)$ candidates we expect to find one that is $L_f(1/2)$-smooth, then solving discrete logarithms also takes $L_f(1/2)$ operations.

Although it looks like we haven't improved over index calculus, the constant hidden in (the exponent of) $L_f(1/2)$ is better for our method—which indeed performs much better in practice—and is the only reason we were able to instantiate parameters twice as large as those of CSI-FiSh (see Section 6).

After having computed a basis for $\mathcal{L}$, the second step, which we do identically to [9], is to apply a lattice reduction algorithm to obtain a shorter basis. Here we need to strike a balance between the time spent reducing and the quality of the output: For example, using BKZ with block size in $O(\sqrt{n})$, running in time $L_{\exp(n)}(1/2)$, we achieve an approximation factor of $L_{\exp(n)}(1/2)$ (see [42, Theorem 3]). In practice, however, the lattice rank n tends to be relatively small, letting us compute a nearly optimal basis in negligible time, as already observed in [9].

[15] This set contains the ideals $\mathfrak{l}_1, \ldots, \mathfrak{l}_m$ that divide $\mathfrak{O}_0\alpha$, but can be larger in general.

Finally, any time we are given an ideal class, say $\mathfrak{l}_1^e$, we use Babai's nearest plane algorithm [4] to find a vector $\boldsymbol{v}$ close to $\boldsymbol{e} = (e, 0, \ldots, 0)$, whence we deduce a representative $\mathfrak{l}_1^{e-v_1} \prod_{i=2}^n \mathfrak{l}_i^{-v_i} \equiv \mathfrak{l}_1^e$. The cost of evaluating the group action by this representative, using the algorithms of Section 3, will be proportional to the norm of $\boldsymbol{e} - \boldsymbol{v}$. Hence the better the basis of $\mathcal{L}$ has been reduced, the faster the evaluation will be.

Choice of prime characteristic. When it comes to the choice of p, we want to find a prime that maximizes the efficiency of evaluating the group action. We have two requirements: that the effective orientations (E, P_E, Q_E) (see Definition 6) can be manipulated efficiently, and that the isogenies associated to the ideals $\mathfrak{l}_i$ can be evaluated efficiently.

For the first requirement, we will force the points P_E and Q_E representing the kernel of $\omega_E = \iota_E(\alpha)$ to be defined over $\mathbb{F}_{p^2}$. Recall that P_E has order $L_1 L_2$ and Q_E has order L_1, hence it is sufficient to choose $L_1 L_2 \mid (p^2 - 1)$.

Similarly, for the second requirement, we want each of the $E[\mathfrak{l}_i]$ to be defined over $\mathbb{F}_{p^2}$ in order to apply the most efficient versions of Vélu's formulas, i.e. we want $n(\mathfrak{l}_i) = \ell_i \mid (p^2 - 1)$. Point in case, $\ell_1, \ldots, \ell_m$ must already divide $p^2 - 1$. Write $L = L_1 L_2 L_3 = \prod_{i=1}^n \ell_i$, then it suffices to select $p = cL \pm 1$ for some small cofactor c.

Finally we want that $\mathcal{S}_{\mathfrak{O}_0}(p)$ is not empty, implying that p must not split in $\mathfrak{O}_0$. For instance, if $\mathfrak{O}_0 = \mathbb{Z}[i]$, we need $p \equiv 3 \pmod 4$. In any case, finding such a prime p can be done after a logarithmic number of tries for a cofactor c. Alternatively, one might take $c = 1$ and play with the split prime factors dividing $L_1 L_2 L_3$ until $L \pm 1$ is prime and split in $\mathfrak{O}_0$.

In fact, we also need p to be large enough to prevent generic attacks (see Section 4). Luckily, with the choices outlined above, we will obtain a prime p that is a lot larger than the minimal security requirement.

Generation of a starting curve. After we have chosen parameters $\mathfrak{O}_0$, L, α, f, p, generated and reduced the lattice of relations $\mathcal{L}$, the last step of precomputation is the generation of a first orientation (E, ι_E) in $\mathcal{S}_{\mathfrak{O}}(p)$. After this last part is done, we will be able to do everything with the group action algorithm. This algorithm will be described later in full detail as Algorithm 3, but for now, we focus on the computation of one (E, ι_E) with the corresponding embedding $\omega_E = \iota_E(\alpha)$. The goal of this paragraph is to explain how the algorithm SetUpCurve works (see Algorithm 1).

First, let us take $(E_0, \iota_0) \in \mathcal{S}_{\mathfrak{O}_0}(p)$, and $\mathcal{O}_0 \cong \mathrm{End}(E_0)$ a maximal order in $B_{p,\infty}$. When d_0 is small enough, $\mathcal{O}_0$ is a special extremal order as defined in [37]. This means that we can efficiently find elements $\gamma \in \mathcal{O}_0$ of norm M as soon as $M > p$. For instance, when $p \equiv 3 \pmod 4$, we can do this in the endomorphism ring of the curve of j-invariant 1728 with the FullRepresentInteger algorithm from [27, Algorithm 1]. Moreover, we can evaluate any endomorphism of $\mathrm{End}(E_0)$ efficiently, because we have the nice representation explicited at the end of Section 2.1. By a result from [43], the orientations in $\mathcal{S}_{\mathfrak{O}}(p)$ are obtained

from the orientations of $\mathcal{S}_{\mathfrak{O}_0}(p)$ through f-isogenies, this is what we explain in Proposition 16.

Proposition 16. *Let $\mathfrak{O}_0$ be a quadratic order, and $(E_0, \iota_0) \in \mathcal{S}_{\mathfrak{O}_0}(p)$, let f be a prime integer and $\mathfrak{O} = \mathbb{Z} + f\mathfrak{O}_0$. If $\varphi : E_0 \to E$ is not one of the $1+(\frac{d_0}{f})$ isogenies corresponding to prime ideals above f, then there exists $\iota_E : \mathfrak{O} \hookrightarrow \mathrm{End}(E)$ and $(E, \iota_E) \in \mathcal{S}_{\mathfrak{O}}(p)$. Moreover $\iota_E(\alpha) = [\varphi]_* \iota_0(\alpha)$ for any $\alpha \in \mathfrak{O}$.*

Now the idea is to compute the kernel of $\iota_0(\alpha)$ (in fact the kernel of the two isogenies of degree L in the decomposition of $\iota_0(\alpha)$) and push that kernel through the isogeny φ. Let us write this kernel as G. The only problem is that in our case f is a large prime, ruling out Vélu's formulas for evaluating φ. Since we know $\mathrm{End}(E_0)$, our idea is to use the method described in [40, Algorithm 2] (or [33]) to evaluate isogenies of large prime degree: represent φ as an ideal I_φ of norm f and compute $J \sim I_\varphi$ where $S = n(J)$, is smooth. Then, evaluate φ, using ψ the isogeny corresponding to J. This is also similar to the key generation of the SQISign signature protocol [26]. Here, we can even use the alternative key generation method described in [26, Appendix D] for better efficiency. Indeed, we can choose almost any isogeny of degree f (by Proposition 16, there are at most two isogenies of degree f that do not create a $\mathfrak{O}$-orientation). Thus, we need to find an endomorphism of norm fS for some smooth integer S. Of course, the simplest situation would be to take $S = 1$, but this is not possible because $f \approx L_1\sqrt{L_2}$ is strictly smaller than p, and we can only find endomorphisms of norm larger than p in $\mathrm{End}(E_0)$. Another natural choice would be to take S dividing L but we need S to be coprime with L_1L_2 because our goal is to evaluate the isogeny of degree S on the L_1L_2-torsion to compute the kernel representation of ω_E. Thus, we can use only the L_3-torsion which is not enough in itself because $fL_3 < p$. We are not going to assume anything specific about the cofactor c (defined along with the prime p as $p = cL \pm 1$), in particular c might not be coprime to L so we may not be able to use it in S. However, c quantifies the size of the additional torsion we need, since we have $c\sqrt{L_2} \approx p/(fL_3)$. What we know for sure is that c is small. Thus, if L_2 is small as well, we can select a small prime ℓ_0 coprime with L_1L_2 and take $S = L_3\ell_0^h$ for some h such that $\ell_0^h > p/(fL_3)$. Since h and ℓ_0 are small, we can simply brute-force through all ℓ_0^h-isogenies until one works, i.e., until we obtain an endomorphism of the right norm and trace after pushing the kernel representation through the considered isogeny of degree ℓ_0^h.

This yields the SetUpCurve algorithm that we describe below as Algorithm 1. The orientation $(E_0, \iota_0) \in \mathcal{S}_{\mathfrak{O}_0}(p)$, and an explicit isomorphism $\rho_0 : \mathcal{O}_0 \hookrightarrow \mathrm{End}(E_0)$ are considered as implicit parameters of this algorithm. The output is a kernel representation of $\iota_E\omega_E$ as in Definition 6.

For a kernel representation s and any morphism ψ, we write $\psi(s)$ for the kernel representation of the group obtained by pushing through ψ the kernel corresponding to s.

Proposition 17. SetUpCurve *is correct and terminates in $O(c\sqrt{L_2}\mathsf{poly}(\log(pcL_2))$ where c is one of $(p \pm 1)/L$.*

Algorithm 1 SetUpCurve(p, f)

Input: A prime p of the form $p = cL_1L_2L_3 \pm 1$ and a prime f such that there exists $\mathfrak{O}_0$ of discriminant d_0 where p is not split and $\mathfrak{O} = \mathbb{Z} + f\mathfrak{O}_0$ contains an element of norm $L_1^2L_2$.

Output: An effective orientation s_E for $(E, \iota_E) \in \mathcal{S}_{\mathfrak{O}}(p)$.

1: Let ℓ_0 be the smallest prime coprime with L_1L_2.
2: Compute s_0 the kernel representation of $\omega_0 = \iota_0(\alpha)$.
3: Set h such that $\ell_0^h > p/(fL_3)$ and compute $\gamma \in \mathcal{O}_0$ of norm $fL_3\ell_0^h$ with FullRepresentInteger.
4: Compute the kernel representation $s = \rho_0(\gamma)(s_0)$.
5: Use ρ_0 to compute the isogeny $\psi : E_0 \to E'$ of norm L_3 corresponding to the ideal $\langle\overline{\gamma}, L_3\rangle$.
6: Make the list $(\varphi_i : E' \to E_i)_{1\leq i\leq m}$ of all isogenies of degree ℓ_0^h from E'.
7: **for** $i \in [1, m]$: **do**
8: Compute $s_i = \varphi_i \circ \psi(s)$ and verify that it is a kernel representation for an endomorphism ω_i of norm $n(\alpha)$ and that it is not s_0.
9: If yes, verify that $\mathrm{tr}(\omega_i)$ is the same as $\mathrm{tr}(\alpha)$. If yes, break from the loop.
10: **end for**
11: Set $E = E_i$, and $s_E = s_i$.
12: **return** Output s_E.

Proof. To prove correctness, we need to verify that the output s_E is an effective orientation in $\mathcal{S}_{\mathfrak{O}}(p)$. Let us assume that the verification made in the loop passed. We will start by proving correctness under that assumption, then we will justify why the verification always passes. When the verification passes, it means that s_E is the kernel representation for an endomorphism ω_E of the same norm and trace as α. This implies that $\mathbb{Z}[\omega_E] \cong \mathbb{Z}[\alpha]$ and so by definition we get that s_E is a correct effective orientation.

Now, let us justify that there always is an i that passes the verification. The element $\gamma \in \mathcal{O}_0$ provides us with a principal ideal $\mathcal{O}_0\gamma$, whose corresponding isogeny $\rho_0(\varphi_\gamma)$ is an endomorphism of E_0. Moreover, we have that (up to composing with some isomorphisms if necessary) $\varphi_\gamma = \psi' \circ \varphi \circ \varphi_f$ where $\varphi_f : E_0 \to E$ has degree f, $\varphi : E \to E'$ has degree ℓ_0^h and $\psi' : E' \to E_0$ has degree L_3. By Proposition 16, E is an $\mathfrak{O}$-orientable curve unless φ_f corresponds to one of the $1 + \left(\frac{d_0}{f}\right)$ horizontal f-isogenies of domain E_0. Let us assume for now that it is not. By Proposition 16, we know that the endomorphism $\omega_E = \iota_E(\alpha)$ can be obtained by pushing forward ω_0 through φ_f. Thus, we need to show that $s = \varphi_f(s_0)$. By design, the ideal $\langle\overline{\gamma}, L_3\rangle$ corresponds to the isogeny $\hat{\psi}'$. Thus, we have that the isogeny ψ computed in Step 5, is the isogeny $\hat{\psi}'$. Then, if we take the index i_0 such that $\varphi_{i_0} = \hat{\varphi}$, we get that E_{i_0} is the curve E that we are looking for. Then, $s_{i_0} = \varphi_{i_0} \circ \psi \circ \psi' \circ \varphi \circ \phi_f(s) = \varphi_f(s)$ and this proves the result. To finish the proof of correctness, we simply need to address the case where φ_f might be one of the bad isogenies. What happens in that case, is that $[\varphi_f]_*\iota_0(\alpha) = \iota_0(\alpha)$ (so we obtain an embedding that is not primitive, since it is the corresponding to ι_0). Thus, the additional verification that s_i is not s_0

prevents the bad case from happening and so we know that s_E is an effective orientation of $\mathcal{S}_{\mathfrak{O}}(p)$.

Regarding complexity, we have $\ell_0^h < \ell_0 p/(fL_3)$ and since we have $f = O(L_1\sqrt{L_2})$, the loop is repeated at most $O(c\sqrt{L_2})$ times. The computations over the quaternions are in $O(\mathsf{poly}(\log(p))$. Then, since we have the explicit isomorphism ρ_0, we can compute ψ and evaluate $\rho_0(\gamma)$ over the L-torsion in $O(\mathsf{poly}(\log(p))$ (remember that the L-torsion is defined over $\mathbb{F}_{p^2}$ and $L < p$). Then, the computation of each φ_i is in $O(\mathsf{poly}(\log(pL_2c))$ and computing s_i and checking the trace has $O(\mathsf{poly}(\log(p))$ complexity with the $\mathsf{CheckTrace}$ algorithm introduced in [40]. This proves the result. □

5.2 The group action computation

Now that we have our starting curve E and an effective orientation ω_E, it remains to see how we can compute $E_{\mathfrak{a}}$ and the kernel representation of $\omega_{E_{\mathfrak{a}}}$ for any ideal $\mathfrak{a}$. For efficiency reasons, we restrict ourselves to the case where $\mathfrak{a}$ has a smooth norm. Also, we target the case where $n(\mathfrak{a}) = \prod_{i=1}^{n} \ell_i^{e_i}$ because this is the one where we will be able to compute the corresponding isogeny efficiently.

Since we only have the L-torsion available, we can factor $\mathfrak{a}$ as the product of $e = \max_{1\leq i\leq n} e_i$ ideals whose norm is dividing L and treat each of them independently.

Thus, our main algorithm is $\mathsf{GroupActionSmall}$ (Algorithm 2) that performs the group action computation for one ideal of degree dividing L. The final algorithm $\mathsf{GroupAction}$ (described as Algorithm 3) is simply the consecutive execution of this sub-algorithm on all factors.

When the ideal has degree dividing L. Let us fix some notation. We write $\mathfrak{L}_1 = \prod_{i=1}^{n_1} \mathfrak{l}_i$, $\mathfrak{L}_2 = \prod_{i=n_1+1}^{n_2} \mathfrak{l}_i$ and $\mathfrak{L}_3 = \prod_{i=n_2+1}^{n} \mathfrak{l}_i$. With these definitions we have $\mathfrak{O}\alpha = \mathfrak{L}_1^2\mathfrak{L}_2$. Equivalently, this means that we can write ω_E as $\hat{\varphi}^E_{\mathfrak{L}_1^{-1}} \circ \varphi^E_{\mathfrak{L}_1\mathfrak{L}_2}$. The kernel of ω_E is made of two subgroups that we write $\langle P_E\rangle, \langle Q_E\rangle$ with $\langle P_E\rangle = \ker \varphi^E_{\mathfrak{L}_1\mathfrak{L}_2}$ and $\langle Q_E\rangle = \ker \varphi^E_{\mathfrak{L}_1^{-1}}$. Let us take the input ideal $\mathfrak{a}$, it can be factored as $\mathfrak{a}_1, \mathfrak{a}_2, \mathfrak{a}_3$ where $n(\mathfrak{a}_i)|L_i$. And for $i = 1, 2$ we also factor $\mathfrak{a}_i$ as $\mathfrak{b}_i\mathfrak{c}_i$ where $\mathfrak{b}_i|\mathfrak{L}_i$ and $\mathfrak{c}_i|\mathfrak{L}_i^{-1}$ and $\gcd(n(\mathfrak{b}_i), n(\mathfrak{c}_i)) = 1$. We write $\mathfrak{K}_i = \mathfrak{L}_i/\mathfrak{b}_i$ and $\mathfrak{J}_1 = \mathfrak{L}_1^{-1}/\mathfrak{c}_1$. Given an ideal $\mathfrak{a}$ whose norm divides L, we use Algorithm 2 ($\mathsf{GroupActionSmall}$) to compute the action of $\mathfrak{a}$ on (E, s_E).

Fig. 2 provides a visualization of the different isogenies involved in Algorithm 2.

Proposition 18. $\mathsf{GroupActionSmall}$ *is correct and runs in time $\tilde{O}(B)$ where B is the largest factor of L.*

Proof. To prove correctness, we need to verify that $s_{E_{\mathfrak{a}}} = (P_{E_{\mathfrak{a}}}, Q_{E_{\mathfrak{a}}})$ represents the two correct subgroups, that is $E_{\mathfrak{a}}[\mathfrak{L}_1\mathfrak{L}_2] = \langle P_{E_{\mathfrak{a}}}\rangle$ and $E_{\mathfrak{a}}[\mathfrak{L}_1^{-1}] = \langle Q_{E_{\mathfrak{a}}}\rangle$. By definition of the effective orientation, we have $E[\mathfrak{L}_1\mathfrak{L}_2] = \langle P_E\rangle$ and $E[\mathfrak{L}_1^{-1}] = \langle Q_E\rangle$.

Algorithm 2 $\mathsf{GroupActionSmall}((E, \iota_E), \mathfrak{a})$

Input: An effective $\mathfrak{O}$-orientation s_E for (E, ι_E) and an ideal $\mathfrak{a} = \mathfrak{b}_1\mathfrak{b}_2\mathfrak{c}_1\mathfrak{c}_2\mathfrak{a}_3$ such that $\mathfrak{b}_i | \mathfrak{L}_i$ and $\mathfrak{c}_i | \mathfrak{L}_i^{-1}$ for $i = 1, 2$ and $n(\mathfrak{a}_3) | L_3$.

Output: An effective $\mathfrak{O}$-orientation $s_{E_\mathfrak{a}}$ for $(E_\mathfrak{a}, \iota_{E_\mathfrak{a}})$.

1: Parse s_E as E, P_E, Q_E.
2: Compute $\varphi^E_{\mathfrak{b}_1\mathfrak{b}_2}$ from its kernel $\langle \left[\frac{L_1L_2}{n(\mathfrak{b}_1\mathfrak{b}_2)}\right] P_E \rangle$
3: Compute $P^*_{E_{\mathfrak{b}_1\mathfrak{b}_2}} = \varphi^E_{\mathfrak{b}_1\mathfrak{b}_2}(P_E)$, $Q_{E_{\mathfrak{b}_1\mathfrak{b}_2}} = \varphi^E_{\mathfrak{b}_1\mathfrak{b}_2}(Q_E)$ and $\varphi^E_{\mathfrak{b}_1\mathfrak{b}_2}(E[n(\mathfrak{c}_2)L_3])$.
4: Compute $\varphi^{E_{\mathfrak{b}_1\mathfrak{b}_2}}_{\mathfrak{K}_1\mathfrak{K}_2}$ from its kernel $\langle P^*_{E_{\mathfrak{b}_1\mathfrak{b}_2}} \rangle$.
5: Compute $Q_{E_{\mathfrak{L}_1\mathfrak{L}_2}} = \varphi^{E_{\mathfrak{b}_1\mathfrak{b}_2}}_{\mathfrak{K}_1\mathfrak{K}_2}(Q_{E_{\mathfrak{b}_1\mathfrak{b}_2}})$ and $\varphi^{E_{\mathfrak{b}_1\mathfrak{b}_2}}_{\mathfrak{K}_1\mathfrak{K}_2}(E_{\mathfrak{b}_1\mathfrak{b}_2}[n(\mathfrak{b}_1\mathfrak{b}_2\mathfrak{c}_2)L_3])$.
6: Compute $\varphi^E_{\mathfrak{c}_1}$ from its kernel $\langle [\frac{L_1}{n(\mathfrak{c}_1)}] Q_E \rangle$
7: Compute $P_{E_{\mathfrak{c}_1}} = \varphi^E_{\mathfrak{c}_1}(P_E)$, $Q^*_{E_{\mathfrak{c}_1}} = \varphi^E_{\mathfrak{c}_1}(Q_E)$ and $\varphi^E_{\mathfrak{c}_1}(E[n(\mathfrak{c}_2)L_3])$.
8: Compute $\varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{J}_1}$ from its kernel $\langle Q^*_{E_{\mathfrak{c}_1}} \rangle$
9: Compute $P_{E_{\mathfrak{L}_1\mathfrak{L}_2}} = \varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{J}_1}(P_{E_{\mathfrak{c}_1}})$ and $\varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{J}_1}(E[n(\mathfrak{c}_1\mathfrak{c}_2)L_3])$.
10: From the action of $\varphi^{E_{\mathfrak{b}_1\mathfrak{b}_2}}_{\mathfrak{K}_1\mathfrak{K}_2}$ on $E_{\mathfrak{b}_1\mathfrak{b}_2}[n(\mathfrak{b}_1\mathfrak{b}_2)]$, compute $\hat{\varphi}^{E_{\mathfrak{b}_1\mathfrak{b}_2}}_{\mathfrak{K}_1\mathfrak{K}_2}([\frac{L_1L_2}{n(\mathfrak{b}_1\mathfrak{b}_2)}] P_{E_{\mathfrak{L}_1\mathfrak{L}_2}})$ and add it up to $P^*_{E_{\mathfrak{b}_1\mathfrak{b}_2}}$ to recover $P_{E_{\mathfrak{b}_1\mathfrak{b}_2}}$.
11: From the action of $\varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{J}_1}$ on $E_{\mathfrak{c}_1}[n(\mathfrak{c}_1)]$, compute $\hat{\varphi}^{E_{\mathfrak{c}_1}}_{\mathfrak{J}_1}([\frac{L_1}{n(\mathfrak{c}_1)}] Q_{E_{\mathfrak{L}_1\mathfrak{L}_2}})$ and add it up to $Q^*_{E_{\mathfrak{c}_1}}$ to recover $Q_{E_{\mathfrak{c}_1}}$.
12: From the action of $\varphi^E_{\mathfrak{b}_1\mathfrak{b}_2}$, $\varphi^{E_{\mathfrak{b}_1\mathfrak{b}_2}}_{\mathfrak{K}_1\mathfrak{K}_2}$, $\varphi^E_{\mathfrak{c}_1}$ and $\varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{J}_1}$ on the respective $n(\mathfrak{c}_2)L_3$-torsion groups, compute $\omega_{E_{\mathfrak{b}_1\mathfrak{b}_2}}(E_{\mathfrak{b}_1\mathfrak{b}_2}[n(\mathfrak{c}_2)L_3])$ and deduce $E_{\mathfrak{b}_1\mathfrak{b}_2}[\mathfrak{c}_2\mathfrak{a}_3]$.
13: Compute $\varphi^{E_{\mathfrak{b}_1\mathfrak{b}_2}}_{\mathfrak{c}_1}$ from its kernel $\langle \left[\frac{L_1}{n(\mathfrak{c}_1)}\right] Q_{E_{\mathfrak{b}_1\mathfrak{b}_2}} \rangle$
14: Compute $P_{E_{\mathfrak{a}_1\mathfrak{b}_2}} = \varphi^{E_{\mathfrak{b}_1\mathfrak{b}_2}}_{\mathfrak{a}_1\mathfrak{b}_2}(P_{E_{\mathfrak{b}_1\mathfrak{b}_2}})$ and $E_{\mathfrak{a}_1\mathfrak{b}_2}[\mathfrak{c}_2\mathfrak{a}_3] = \varphi^{E_{\mathfrak{b}_1\mathfrak{b}_2}}_{\mathfrak{c}_1}(E_{\mathfrak{b}_1\mathfrak{b}_2}[\mathfrak{c}_2\mathfrak{a}_3])$.
15: Compute $\varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{b}_1\mathfrak{b}_2}$ from its kernel $\langle \left[\frac{L_1L_2}{n(\mathfrak{b}_1\mathfrak{b}_2)}\right] P_{E_{\mathfrak{c}_1}} \rangle$
16: Compute $Q_{E_{\mathfrak{a}_1\mathfrak{b}_2}} = \varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{b}_1\mathfrak{b}_2}(Q_{E_{\mathfrak{c}_1}})$.
17: Compute $\varphi^{E_{\mathfrak{a}_1\mathfrak{b}_2}}_{\mathfrak{c}_2\mathfrak{a}_3} = \varphi^{E_{\mathfrak{a}_1\mathfrak{a}_2}}_{\mathfrak{a}_3} \circ \varphi^{E_{\mathfrak{a}_1\mathfrak{b}_2}}_{\mathfrak{c}_2}$ from its kernel $E_{\mathfrak{a}_1\mathfrak{b}_2}[\mathfrak{c}_2\mathfrak{a}_3]$.
18: Compute $P_{E_\mathfrak{a}} = \varphi^{E_{\mathfrak{a}_1\mathfrak{b}_2}}_{\mathfrak{c}_2\mathfrak{a}_3}(P_{E_{\mathfrak{a}_1\mathfrak{b}_2}})$ and $Q_{E_\mathfrak{a}} = \varphi^{E_{\mathfrak{a}_1\mathfrak{b}_2}}_{\mathfrak{c}_2\mathfrak{a}_3}(Q_{E_{\mathfrak{a}_1\mathfrak{b}_2}})$.
19: Compute the canonical effective orientation $s_{E_\mathfrak{a}}$ for $(E_\mathfrak{a}, \iota_{E_\mathfrak{a}})$ from $E_\mathfrak{a}$, $P_{E_\mathfrak{a}}$ and $Q_{E_\mathfrak{a}}$ (see Remark 7).
20: **return** $s_{E_\mathfrak{a}}$.

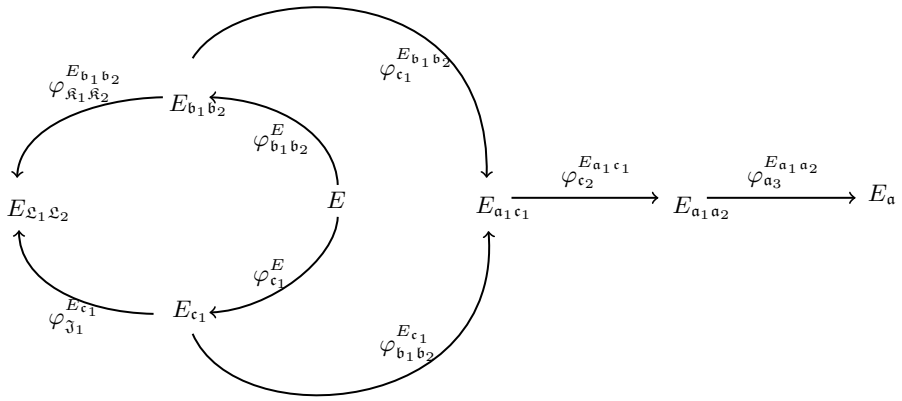

Fig. 2. A picture of the isogenies and curves involved in GroupActionSmall.

From the computation of the isogenies $\varphi^E_{\mathfrak{b}_1\mathfrak{b}_2}$, $\varphi^{E_{\mathfrak{b}_1\mathfrak{b}_2}}_{\mathfrak{K}_1\mathfrak{K}_2}$, $\varphi^E_{\mathfrak{c}_1}$ and $\varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{J}_1}$ in step 2, 4, 6 and 8 respectively, and their evaluation on the respective $n(\mathfrak{c}_2\mathfrak{a}_3)$ torsion groups in step 3, 5, 7 and 9, we successfully recover the action of

$$\omega_{E_{\mathfrak{b}_1\mathfrak{b}_2}} = \varphi^E_{\mathfrak{b}_1\mathfrak{b}_2} \circ \hat{\varphi}^E_{\mathfrak{c}_1} \circ \hat{\varphi}^{E_{\mathfrak{c}_1}}_{\mathfrak{J}_1} \circ \varphi^{E_{\mathfrak{b}_1\mathfrak{b}_2}}_{\mathfrak{K}_1\mathfrak{K}_2}$$

on $E_{\mathfrak{b}_1\mathfrak{b}_2}[n(\mathfrak{c}_2)L_3]$ in step 12. Since $n(\mathfrak{c}_2)L_3$ is smooth, we efficiently solve some two-dimensional discrete logarithms in the group $E_{\mathfrak{b}_1\mathfrak{b}_2}[n(\mathfrak{c}_2)L_3]$ to successfully recover $E_{\mathfrak{b}_1\mathfrak{b}_2}[\mathfrak{c}_2\mathfrak{a}_3]$ in step 12.

Applying Lemma 5 to $(E, \mathfrak{b}_1\mathfrak{b}_2, \mathfrak{L}^{-1})$, we get that $\langle Q_{E_{\mathfrak{b}_1\mathfrak{b}_2}}\rangle = E_{\mathfrak{b}_1\mathfrak{b}_2}[\mathfrak{L}_1^{-1}]$ in step 3. Meanwhile, in step 3 $\langle P^*_{E_{\mathfrak{b}_1\mathfrak{b}_2}}\rangle = \langle\varphi^E_{\mathfrak{b}_1\mathfrak{b}_2}(P_E)\rangle$ generates the proper subgroup of $E_{\mathfrak{b}_1\mathfrak{b}_2}[\mathfrak{L}_1\mathfrak{L}_2]$ of order $L_1L_2/n(\mathfrak{b}_1\mathfrak{b}_2)$.

To recover the remaining part of the group $E_{\mathfrak{b}_1\mathfrak{b}_2}[\mathfrak{L}_1\mathfrak{L}_2]$, one applies the formulas given in Proposition 9: that is, one recovers the part of $E_{\mathfrak{b}_1\mathfrak{b}_2}[\mathfrak{L}_1\mathfrak{L}_2]$ lost when evaluating $\varphi^E_{\mathfrak{b}_1\mathfrak{b}_2}$ on P_E by evaluating

$$\varphi^E_{(\mathfrak{L}_1K_1K_2)^{-1}} = \hat{\varphi}^{E_{\mathfrak{b}_1\mathfrak{b}_2}}_{\mathfrak{K}_1\mathfrak{K}_2} \circ \varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{J}_1} \circ \varphi^E_{\mathfrak{c}_1}$$

on $[\frac{L_1L_2}{n(\mathfrak{b}_1\mathfrak{b}_2)}]P_E$. This is done in step 10 where $E_{\mathfrak{b}_1\mathfrak{b}_2}[\mathfrak{L}_1\mathfrak{L}_2] = \langle P_{E_{\mathfrak{b}_1\mathfrak{b}_2}}\rangle$.

Reasoning similarly for $\mathfrak{c}_1$ and $\mathfrak{L}_1\mathfrak{L}_2$, we get that in step 7, we have the equality $\langle P_{E_{\mathfrak{c}_1}}\rangle = E_{\mathfrak{c}_1}[\mathfrak{L}_1\mathfrak{L}_2]$ and that step 11 successfully recovers $Q_{E_{\mathfrak{c}_1}}$ such that $E_{\mathfrak{c}_1}[\mathfrak{L}_1^{-1}] = \langle Q_{E_{\mathfrak{c}_1}}\rangle$.

Applying Lemma 5 to $(E_{\mathfrak{c}_1}, \mathfrak{b}_1\mathfrak{b}_2, \mathfrak{L}^{-1})$, $(E_{\mathfrak{b}_1\mathfrak{b}_2}, \mathfrak{c}_1, \mathfrak{L}_1\mathfrak{L}_2)$ and $(E_{\mathfrak{b}_1\mathfrak{b}_2}, \mathfrak{c}_1, \mathfrak{c}_2\mathfrak{a}_3)$ respectively, we get that

$$E_{\mathfrak{a}_1\mathfrak{b}_2}[\mathfrak{L}_1^{-1}] = \varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{b}_1\mathfrak{b}_2}\left(E_{\mathfrak{c}_1}[\mathfrak{L}_1^{-1}]\right) = \varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{b}_1\mathfrak{b}_2}(\langle Q_{E_{\mathfrak{c}_1}}\rangle) = \langle Q_{E_{\mathfrak{a}_1\mathfrak{b}_2}}\rangle$$

Algorithm 3 GroupAction$((E, \iota_E), \mathfrak{d})$

Input: An effective $\mathfrak{O}$-orientation s_E for (E, ι_E) and $\mathfrak{d} = \mathfrak{l}_1^{e_1} \cdots \mathfrak{l}_n^{e_n}$.
Output: An effective $\mathfrak{O}$-orientation $s_{E_\mathfrak{d}}$ for $(E_\mathfrak{d}, \iota_{E_\mathfrak{d}})$

while some $e_i \neq 0$ **do**
  $\mathfrak{a} = 1$
  for $i \in \{1, \cdots, n\}$ **do**
    if $e_i < 0$ **then**
      $\mathfrak{a} = \mathfrak{a} * \mathfrak{l}_i^{-1}$, $e_i = e_i + 1$
    else if $e_i > 0$ **then**
      $\mathfrak{a} = \mathfrak{a} * \mathfrak{l}_i$, $e_i = e_i - 1$
    end if
  end for
  $s_E =$ GroupActionSmall$(s_E, \mathfrak{a})$
end while
return s_E.

as computed in step 16,

$$E_{\mathfrak{a}_1\mathfrak{b}_2}[\mathfrak{L}_1\mathfrak{L}_2] = \varphi_{\mathfrak{c}_1}^{E_{\mathfrak{b}_1\mathfrak{b}_2}}(E_{\mathfrak{b}_1\mathfrak{b}_2}[\mathfrak{L}_1\mathfrak{L}_2]) = \varphi_{\mathfrak{c}_1}^{E_{\mathfrak{b}_1\mathfrak{b}_2}}(\langle P_{E_{\mathfrak{b}_1\mathfrak{b}_2}}\rangle) = \langle P_{E_{\mathfrak{a}_1\mathfrak{b}_2}}\rangle$$

as computed in step 14 and

$$E_{\mathfrak{a}_1\mathfrak{b}_2}[\mathfrak{c}_2\mathfrak{a}_3] = \varphi_{\mathfrak{c}_1}^{E_{\mathfrak{b}_1\mathfrak{b}_2}}(E_{\mathfrak{b}_1\mathfrak{b}_2}[\mathfrak{c}_2\mathfrak{a}_3])$$

as computed in step 14.

In step 17 and 18, we compute $\varphi_{\mathfrak{c}_2\mathfrak{a}_3}^{E_{\mathfrak{a}_1\mathfrak{b}_2}}$ and applying Lemma 5 to $(E_{\mathfrak{a}_1\mathfrak{b}_2}, \mathfrak{c}_2\mathfrak{a}_3, \mathfrak{L}^{-1})$ and $(E_{\mathfrak{a}_1\mathfrak{b}_2}, \mathfrak{c}_2\mathfrak{a}_3, \mathfrak{L}_1\mathfrak{L}_2)$ respectively, we get

$$E_\mathfrak{a}[\mathfrak{L}_1^{-1}] = \varphi_{\mathfrak{c}_2\mathfrak{a}_3}^{E_{\mathfrak{a}_1\mathfrak{b}_2}}(E_{\mathfrak{a}_1\mathfrak{b}_1}[\mathfrak{L}_1^{-1}]) = \varphi_{\mathfrak{c}_2\mathfrak{a}_3}^{E_{\mathfrak{a}_1\mathfrak{b}_2}}(\langle Q_{\mathfrak{a}_1\mathfrak{b}_1}\rangle) = \langle Q_{E_\mathfrak{a}}\rangle$$

and

$$E_\mathfrak{a}[\mathfrak{L}_1\mathfrak{L}_2] = \varphi_{\mathfrak{c}_2\mathfrak{a}_3}^{E_{\mathfrak{a}_1\mathfrak{b}_2}}(E_{\mathfrak{a}_1\mathfrak{b}_1}[\mathfrak{L}_1\mathfrak{L}_2]) = \varphi_{\mathfrak{c}_2\mathfrak{a}_3}^{E_{\mathfrak{a}_1\mathfrak{b}_2}}(\langle P_{\mathfrak{a}_1\mathfrak{b}_1}\rangle) = \langle P_{E_\mathfrak{a}}\rangle.$$

Algorithm 2 mostly consists of scalar multiplications, isogenies and discrete logarithm computations. The running time of scalar multiplications is polynomial in $\log(p)$ and $\log(L)$. Since the degrees of the isogenies computed, and the orders of the groups in which the discrete logarithms are computed divide L, then these operations can be performed in time $\tilde{O}(B)$ where B is the largest factor of L. Hence the overall complexity of Algorithm 2, ignoring logarithmic factors, is $\tilde{O}(B)$. □

The full algorithm. Now, Algorithm 3 describes the group action evaluation. It is simply made of consecutive executions of GroupActionSmall preceded with a little initialization.

6 Concrete instantiation

In this section, we report on the concrete choices we made to instantiate a signature scheme analogous to CSI-FiSh on top of our SCALLOP group action.

For the construction of the signature scheme it suffices to replace the CSIDH group action by the SCALLOP group action. Since this does not provide any new insights, we refer the reader to [9] for the detailed description of the scheme instead. The security of the new signature scheme based on the SCALLOP group action relies on the problems introduced in Section 4. For the concrete instantiation we target two levels of security: matching the security of CSIDH-512 and of CSIDH-1024. To obtain class groups of the same size, we take prime conductors of size 256 and 512 bits respectively.

6.1 Parameter selection

As outlined in Section 5, we start by choosing the conductor f. To this end, we fix $\mathfrak{O}_0 = \mathbb{Z}[i]$ to be the Gaussian integers. Then, we consider the smallest $n_1 + n_2$ split primes ℓ_i. As before, let $\mathfrak{l}_i$ denote split ideals associated to the primes ℓ_i. We partition the primes into two sets P_1 and P_2 of respective size n_1 and n_2 such that $L_1 = \prod_{\ell_i \in P_1} \ell_i$ and $L_2 = \prod_{\ell_i \in P_2} \ell_i$. For such a fixed partition, we iterate through choices for $b_i \in \{-2, 2\}$ and $c_i \in \{-1, 1\}$ to generate candidates for the orientation $\alpha \in \mathbb{Z}[i]$ as

$$\prod_{\ell_i \in P_1} \mathfrak{l}_i^{b_i} \prod_{\ell_i \in P_2} \mathfrak{l}_i^{c_i}.$$

By construction, each candidate is of smooth norm $L_1^2 L_2$.

For each candidate, we test whether the coefficient f of the imaginary part is prime. If this is the case, we try to factor $f + 1$, if $f \equiv 3 \pmod 4$, or $f - 1$ otherwise. Factoring is done using the ECM method with early abort in case a factor larger than a given smoothness bound is found or no further factor is discovered within a given time frame.

We ran this method and the algorithm SetUpCurve to find a conductor and a starting oriented curve for parameters with the same security level as CSIDH-512 and CSIDH-1024 respectively. The result are reported in Appendix B. In both cases, the computation ran in minutes on a laptop.

6.2 Performance

Size of public keys. Public keys are represented as effective orientations (E, P_E, Q_E) (see Definition 6), with all constants defined over $\mathbb{F}_{p^2}$, so they are approximately six times larger than CSIDH keys. However, using standard compression techniques, we can represent them using only two $\mathbb{F}_p$-elements and two integers modulo $L_1 L_2$, which would give keys of approximately 1600 bits for SCALLOP-512 and 2300 bits for SCALLOP-1024.

Implementation. We implemented our group action in C++, making use of assembly-language field arithmetic. In our proof-of-concept implementation, applying the action of one arbitrary class-group element takes about 35 seconds for the smaller parameter set and 12.5 minutes for the larger parameter set on a single core of an Intel i5-6440HQ processor running at 3.5 GHz. Note that our implementation is not side-channel resistant.

While the current implementation is not fully optimized, for instance it does not yet use the $\sqrt{\text{élu}}$ algorithm [6], we do not expect to gain an order of magnitude by implementing all the possible optimizations. Thus, even if our implementation demonstrates feasibility, it seems that the SCALLOP group action is not yet ready for cryptographic applications.

Our code is available at `https://github.com/isogeny-scallop/scallop`.

7 Security discussion: evaluating the descending isogeny

We discuss a conceivable strategy to break the hardness assumptions of our proposed group action in the following. Recall that $\mathfrak{O}$-Vectorisation is essentially equivalent to $\mathfrak{O}$-EndRing, hence it would be sufficient to devise an algorithm that computes the endomorphism ring of any $\mathfrak{O}$-oriented curve, say (E_1, ι_1). Given an $\mathfrak{O}_0$-oriented curve (E_0, ι_0) with known endomorphism ring and $\mathfrak{O}_0$ of class number one, there exists a unique descending isogeny

$$\varphi : (E_0, \iota_0) \longrightarrow (E_1, \iota_1),$$

which has degree f. To compute $\mathrm{End}(E_1)$, one could try the following:

1. Find an algorithm to evaluate φ on input points efficiently.
2. Using Step 1, try to convert φ into its corresponding left $\mathrm{End}(E_0)$-ideal I_φ.
3. Deduce $\mathrm{End}(E_1)$ as the right-order of I_φ.

Note that this problem is related to the SubOrder to Ideal Problem (SOIP) introduced by Leroux [40]. It is quite obvious that the problem we study here is harder than the SOIP since the SOIP provides to the attacker several effective orientations of different quadratic orders (instead of one in our case). We refer to [40, section 4] for a study of the SOIP. Below, we will try to explain why applying efficiently the attack outlined above appears complicated.

In particular, the first two steps seem challenging. Since we chose $\deg(\varphi) = f$ to be a large prime, there is no hope to evaluate φ, Step 1, using standard algorithms such as Vélu's formulas, which have polynomial complexity in $\deg(\varphi)$. However, even if one managed to solve Step 1, it is not clear how to solve Step 2 (which is somewhat equivalent to the SOIP, see [40, Proposition 14]). Known algorithms to convert an isogeny into an ideal require working within the torsion subgroup $E[\deg(\varphi)]$. Our parameter choice ensures this torsion to be defined over an extension field of exponentially large degree.

Despite these obstacles, let us investigate a possible solution to Step 1, which does not necessarily need to rely on Vélu's formulas, or knowing $\ker(\varphi)$.

Let us introduce a vector notation for arithmetic on the curves. Given a pair of points $B = (P, Q)$, and a vector of two integers $v = (x, y)$, we write $v \cdot B = xP + yQ$. Fix a positive integer n coprime with p and the norm of $\mathfrak{a}$. Let $B_0 = (P_0, Q_0)$ and $B_1 = (P_1, Q_1)$ be bases of $E_0[n]$ and $E_1[n]$ respectively. Let $\psi : E_0 \to E_1$ be an isogeny. The restriction of ψ on the n-torsion is characterised by the matrix $M_\psi \in M_{2\times 2}(\mathbb{Z}/n\mathbb{Z})$ such that for any $v \in (\mathbb{Z}/n\mathbb{Z})^2$, we have $\psi(v \cdot B_0) = (M_\psi v) \cdot B_1$. We call M_ψ the matrix form of ψ with respect to B_0 and B_1.

In the following, we show that even for φ of large prime degree, it is possible to learn information about M_φ, effectively identifying a 1-dimensional subvariety of $M_{2\times 2}(\mathbb{Z}/n\mathbb{Z})$ containing it. Yet, this is not enough to solve Step 1.

Let $e_n(-,-)$ denote the Weil pairing on points of order dividing n. The following lemma fixes the determinant of M_φ.

Lemma 19. *If $e_n(P_0, Q_0) = e_n(P_1, Q_1)$, then $\det(M_\varphi) \equiv \deg(\varphi) \bmod n$.*

Proof. Write $M_\varphi = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$. We have

$$\begin{aligned} e_n(P_0, Q_0)^{\deg(\varphi)} &= e_n(\varphi(P_0), \varphi(Q_0)) = e_n(aP_1 + cQ_1, bP_1 + dQ_1) \\ &= e_n(P_1, Q_1)^{ad-bc} = e_n(P_0, Q_0)^{\det(M_\varphi)}. \end{aligned}$$

The result follows from the non-degeneracy of the Weil pairing. □

For random bases B_0 and B_1, $e_n(P_0, Q_0) = e_n(P_1, Q_1)$ is unlikely. However, at the cost of solving one discrete logarithm in a group of order n, this condition on the bases can be enforced. This can be done in classical exponential time in the size of the largest prime factor of n, or in quantum polynomial time in $\log(n)$.

Due to φ being a descending isogeny, we observe that M_φ satisfies further certain linear relations: Writing $\mathfrak{O}_0 = \mathbb{Z}[\omega]$ and $\mathfrak{O} = \mathbb{Z}[f\omega]$, we have $\iota_1(f\omega) = \varphi \circ \iota_0(\omega) \circ \hat{\varphi}$, hence

$$AM_\varphi = M_\varphi B$$

where A is the matrix of $\iota_1(f\omega)$ (with respect to B_1), and B is the matrix of $f\iota_0(\omega)$ (with respect to B_0). Note that the matrices A and B can be computed in quantum polynomial time (or in classical exponential time in the size of the largest prime factor). This is because the endomorphisms can be evaluated in polynomial time on the points of the basis, and the matrix coefficients follow from a discrete logarithm computation as above.

For simplicity, assume that n is prime. Then, $M_{2\times 2}(\mathbb{Z}/n\mathbb{Z})$ is an $\mathbb{F}_n$-vector space. The space $\mathcal{M}$ of solutions M of $AM_\varphi = M_\varphi B$ has dimension 2. Indeed, if M is one solution with non-zero determinant, then XM is a solution if and only if X commutes with A. Note that a solution exists, since M_φ itself has non-zero determinant by Lemma 19. The space of matrices that commute with A is the span of A and the identity matrix I_2, which has rank 1 if A is a scalar matrix, and 2 otherwise. Since n is coprime with the norm of $\mathfrak{a}$, the endomorphism $\iota_1(f\omega)$

does not act like a scalar on the n-torsion, so A is not a scalar matrix, and the space of solutions $\mathcal{M}$ has dimension 2.

Together with Lemma 19, we have reduced our search space for M_φ to the one-dimensional $\mathbb{F}_n$-variety

$$\mathcal{M}_f = \{M \in \mathcal{M} | \det(M) = f\}.$$

It is unclear how to reduce this space further, narrowing down M_φ. One may be tempted to use pairing equations as in Lemma 19 with the Tate pairing instead of the Weil pairing. However, the curves having trace $\pm 2p$, the Tate pairing is alternating (see [53, Theorem 3.17]), and thereby provides the same condition as the Weil pairing. In conclusion, it appears that all the available information is insufficient to evaluate the descending isogeny φ on any input efficiently.

References

1. Alamati, N., De Feo, L., Montgomery, H., Patranabis, S.: Cryptographic group actions and applications. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 411–439. Springer (2020)
2. Arpin, S., Chen, M., Lauter, K.E., Scheidler, R., Stange, K.E., Tran, H.T.N.: Orienteering with one endomorphism. arXiv preprint arXiv:2201.11079 (2022)
3. Azarderakhsh, R., Jao, D., Kalach, K., Koziel, B., Leonardi, C.: Key compression for isogeny-based cryptosystems. In: Proceedings of the 3rd ACM International Workshop on ASIA Public-Key Cryptography. pp. 1–10. ACM (2016)
4. Babai, L.: On Lovász' lattice reduction and the nearest lattice point problem. Combinatorica **6**(1), 1–13 (1986)
5. Belding, J.V.: Number theoretic algorithms for elliptic curves. University of Maryland, College Park (2008)
6. Bernstein, D.J., De Feo, L., Leroux, A., Smith, B.: Faster computation of isogenies of large prime degree. ANTS (2020)
7. Beullens, W., Dobson, S., Katsumata, S., Lai, Y.F., Pintore, F.: Group signatures and more from isogenies and lattices: Generic, simple, and efficient. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 95–126. Springer (2022)
8. Beullens, W., Katsumata, S., Pintore, F.: Calamari and Falafl: logarithmic (linkable) ring signatures from isogenies and lattices. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 464–492. Springer (2020)
9. Beullens, W., Kleinjung, T., Vercauteren, F.: CSI-FiSh: Efficient isogeny based signatures through class group computations. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 227–247. Springer (2019)
10. Bonnetain, X., Schrottenloher, A.: Quantum security analysis of CSIDH. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 493–522. Springer (2020)
11. Campos, F., Muth, P.: On actively secure fine-grained access structures from isogeny assumptions. In: Cheon, J.H., Johansson, T. (eds.) Post-Quantum Cryptography. pp. 375–398. Springer International Publishing, Cham (2022). https://doi.org/10.1007/978-3-031-17234-2_18

12. Castryck, W., Houben, M., Vercauteren, F., Wesolowski, B.: On the decisional Diffie-Hellman problem for class group actions on oriented elliptic curves. Research in Number Theory **8** (2022). https://doi.org/10.1007/s40993-022-00399-6
13. Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: CSIDH: an efficient post-quantum commutative group action. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 395–427. Springer (2018)
14. Castryck, W., van der Meeren, N.: Two remarks on the vectorization problem. Cryptology ePrint Archive (2022)
15. Castryck, W., Sotáková, J., Vercauteren, F.: Breaking the decisional Diffie-Hellman problem for class group actions using genus theory. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology – CRYPTO 2020. Lecture Notes in Computer Science, vol. 12171, pp. 92–120. Springer (2020). https://doi.org/10.1007/978-3-030-56880-1_4
16. Chávez-Saab, J., Chi-Domínguez, J.J., Jaques, S., Rodríguez-Henríquez, F.: The SQALE of CSIDH: sublinear Vélu quantum-resistant isogeny action with low exponents. Journal of Cryptographic Engineering **12**(3), 349–368 (2022)
17. Childs, A.M., van Dam, W.: Quantum algorithms for algebraic problems. Reviews of Modern Physics **82**(1), 1 (2010)
18. Chung, K.M., Hsieh, Y.C., Huang, M.Y., Huang, Y.H., Lange, T., Yang, B.Y.: Group signatures and accountable ring signatures from isogeny-based assumptions. arXiv preprint arXiv:2110.04795 (2021)
19. Colò, L., Kohel, D.: Orienting supersingular isogeny graphs. Number-Theoretic Methods in Cryptology 2019 (2019)
20. Costello, C., Jao, D., Longa, P., Naehrig, M., Renes, J., Urbanik, D.: Efficient Compression of SIDH Public Keys, pp. 679–706. Springer International Publishing (2017). https://doi.org/10.1007/978-3-319-56620-7_24
21. Couveignes, J.M.: Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291 (2006)
22. Cozzo, D., Smart, N.P.: Sashimi: cutting up CSI-FiSh secret keys to produce an actively secure distributed signing protocol. In: International Conference on Post-Quantum Cryptography. pp. 169–186. Springer (2020)
23. Dartois, P., De Feo, L.: On the security of OSIDH. In: IACR International Conference on Public-Key Cryptography. pp. 52–81. Springer (2022)
24. De Feo, L., Fouotsa, T.B., de Saint Guilhem, C.D., Kutas, P., Leroux, A., Petit, C., Silva, J., Wesolowski, B.: SÉTA: Supersingular encryption from torsion attacks. In: ASIACRYPT (2021)
25. De Feo, L., Galbraith, S.D.: Seasign: compact isogeny signatures from class group actions. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 759–789. Springer (2019)
26. De Feo, L., Kohel, D., Leroux, A., Petit, C., Wesolowski, B.: SQISign: compact post-quantum signatures from quaternions and isogenies. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 64–93. Springer (2020)
27. De Feo, L., Leroux, A., Longa, P., Wesolowski, B.: New algorithms for the Deuring correspondence: Towards practical and secure sqisign signatures. to appear in Eurocrypt 2023 (2023)
28. De Feo, L., Meyer, M.: Threshold schemes from isogeny assumptions. In: IACR International Conference on Public-Key Cryptography. pp. 187–212. Springer (2020)

29. Decru, T., Panny, L., Vercauteren, F.: Faster seasign signatures through improved rejection sampling. In: International Conference on Post-Quantum Cryptography. pp. 271–285. Springer (2019)
30. Delfs, C., Galbraith, S.D.: Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$. Designs, Codes and Cryptography **78**(2), 425–440 (2016)
31. Eisenträger, K., Hallgren, S., Lauter, K., Morrison, T., Petit, C.: Supersingular isogeny graphs and endomorphism rings: Reductions and solutions. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2018. pp. 329–368. Springer International Publishing (2018)
32. Eisenträger, K., Hallgren, S., Leonardi, C., Morrison, T., Park, J.: Computing endomorphism rings of supersingular elliptic curves and connections to path-finding in isogeny graphs. Open Book Series **4**(1), 215–232 (2020)
33. Fouotsa, T.B., Kutas, P., Merz, S.P., Ti, Y.B.: On the isogeny problem with torsion point information. In: IACR International Conference on Public-Key Cryptography. pp. 142–161. Springer (2022)
34. Galbraith, S., Panny, L., Smith, B., Vercauteren, F.: Quantum equivalence of the DLP and CDHP for group actions. Mathematical Cryptology **1**(1), 40–44 (2021)
35. Hardy, G.H., Wright, E.M., et al.: An introduction to the theory of numbers. Oxford university press (1979)
36. Ivanyos, G.: On solving systems of random linear disequations. arXiv preprint arXiv:0704.2988 (2007)
37. Kohel, D.R., Lauter, K., Petit, C., Tignol, J.P.: On the quaternion ℓ-isogeny path problem. LMS Journal of Computation and Mathematics **17**(A), 418–432 (2014)
38. Kuperberg, G.: A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. SIAM Journal on Computing **35**(1), 170–188 (2005)
39. Lai, Y.F., Dobson, S.: Collusion resistant revocable ring signatures and group signatures from hard homogeneous spaces. Cryptology ePrint Archive (2021)
40. Leroux, A.: A new isogeny representation and applications to cryptography. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology – ASIACRYPT 2022. pp. 3–35. Springer Nature Switzerland, Cham (2022). https://doi.org/10.1007/978-3-031-22966-4_1
41. Leroux, A., Roméas, M.: Updatable encryption from group actions. Cryptology ePrint Archive (2022)
42. Li, J., Nguyen, P.Q.: A complete analysis of the BKZ lattice reduction algorithm. Cryptology ePrint Archive, Paper 2020/1237 (2020), https://eprint.iacr.org/2020/1237
43. Love, J., Boneh, D.: Supersingular curves with small noninteger endomorphisms. Open Book Series **4**(1), 7–22 (2020)
44. Montgomery, H., Zhandry, M.: Full quantum equivalence of group action DLog and CDH, and more. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology – ASIACRYPT 2022. pp. 3–32. Springer Nature Switzerland, Cham (2022). https://doi.org/10.1007/978-3-031-22963-3_1
45. Naehrig, M., Renes, J.: Dual isogenies and their application to public-key compression for isogeny-based cryptography. In: Galbraith, S.D., Moriai, S. (eds.) Advances in Cryptology – ASIACRYPT 2019. pp. 243–272. Springer International Publishing, Cham (2019)
46. Onuki, H.: On oriented supersingular elliptic curves. Finite Fields and Their Applications **69**, 101777 (2021)
47. Peikert, C.: He gives C-sieves on the CSIDH. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 463–492. Springer (2020)

48. de Quehen, V., Kutas, P., Leonardi, C., Martindale, C., Panny, L., Petit, C., Stange, K.E.: Improved torsion-point attacks on SIDH variants. In: Annual International Cryptology Conference. pp. 432–470. Springer (2021)
49. Rostovtsev, A., Stolbunov, A.: Public-key cryptosystem based on isogenies. Cryptology ePrint Archive, Report 2006/145 (2006)
50. Silverman, J.H.: The Arithmetic of Elliptic Curves, vol. 106 (01 2009)
51. Stolbunov, A.: Cryptographic schemes based on isogenies (2012)
52. Vélu, J.: Isogénies entre courbes elliptiques. Comptes-Rendus de l'Académie des Sciences, Série I **273**, 238–241 (juillet 1971)
53. Washington, L.C.: Elliptic curves: number theory and cryptography. Chapman and Hall/CRC, second edn. (2008). https://doi.org/10.1201/9781420071474
54. Wesolowski, B.: Orientations and the supersingular endomorphism ring problem. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology – EUROCRYPT 2022. Lecture Notes in Computer Science, vol. 13277, pp. 345–371. Springer (2022). https://doi.org/10.1007/978-3-031-07082-2_13
55. Zanon, G.H.M., Simplicio, M.A., Pereira, G.C.C.F., Doliskani, J., Barreto, P.S.L.M.: Faster isogeny-based compressed key agreement. In: Lange, T., Steinwandt, R. (eds.) Post-Quantum Cryptography. pp. 248–268. Springer International Publishing (2018)

A Mapping finite fields and class groups

In this section, we investigate isomorphisms between certain ideal class groups and finite fields. Let $\mathfrak{O}_K$ be the ring of integers of a quadratic field K. Write $\mathfrak{O}_K = \mathbb{Z}[X]/f(X)$ and $\omega = X \bmod f(X)$. Let ℓ be a prime number that is not ramified in $\mathfrak{O}_K$, and let $\mathfrak{O} = \mathbb{Z} + \ell\mathfrak{O}_K$. We have the exact sequence

$$1 \longrightarrow \mathfrak{O}_K^\times/\mathfrak{O}^\times \longrightarrow (\mathfrak{O}_K/\ell\mathfrak{O}_K)^\times/(\mathfrak{O}/\ell\mathfrak{O})^\times \longrightarrow \mathrm{Cl}(\mathfrak{O}) \longrightarrow \mathrm{Cl}(\mathfrak{O}_K) \longrightarrow 1.$$

Assuming $\mathrm{Cl}(\mathfrak{O}_K)$ is trivial,

$$1 \longrightarrow \mathfrak{O}_K^\times/\mathfrak{O}^\times \longrightarrow (\mathfrak{O}_K/\ell\mathfrak{O}_K)^\times/(\mathfrak{O}/\ell\mathfrak{O})^\times \longrightarrow \mathrm{Cl}(\mathfrak{O}) \longrightarrow 1.$$

In the following, we describe isomorphisms from $(\mathfrak{O}_K/\ell\mathfrak{O}_K)^\times/(\mathfrak{O}/\ell\mathfrak{O})^\times$ to either $\mathbb{F}_\ell^\times$ or $\mathbb{F}_{\ell^2}^\times/\mathbb{F}_\ell^\times$.

A.1 The split case

Lemma 20. *If ℓ splits in $\mathfrak{O}_K$ as $\mathfrak{l}\bar{\mathfrak{l}}$, the map*

$$(\mathfrak{O}_K/\ell\mathfrak{O}_K)^\times/(\mathfrak{O}/\ell\mathfrak{O})^\times \longrightarrow (\mathfrak{O}_K/\mathfrak{l})^\times : \alpha \longmapsto \alpha/\overline{\alpha}$$

is an isomorphism.

Proof. By the Chinese remainder theorem and the coprimality of $\mathfrak{l}$ and $\bar{\mathfrak{l}}$, the map

$$\chi : (\mathfrak{O}_K/\ell\mathfrak{O}_K)^\times \longrightarrow (\mathfrak{O}_K/\mathfrak{l})^\times \times (\mathfrak{O}_K/\bar{\mathfrak{l}})^\times,\ \alpha \longmapsto (\alpha, \alpha)$$

is a group isomorphism. Furthermore, the map

$$\delta : (\mathfrak{O}_K/\mathfrak{l})^\times \times (\mathfrak{O}_K/\bar{\mathfrak{l}})^\times \longrightarrow (\mathfrak{O}_K/\mathfrak{l})^\times,\ (\alpha,\beta) \longmapsto \alpha/\overline{\beta}$$

is a well-defined, surjective homomorphism. We get a surjective homomorphism

$$\delta \circ \chi : (\mathfrak{O}_K/\ell\mathfrak{O}_K)^\times \longrightarrow (\mathfrak{O}_K/\mathfrak{l})^\times,\ \alpha \longmapsto \alpha/\overline{\alpha}.$$

It is easy to verify that its kernel is $(\mathfrak{O}/\ell\mathfrak{O})^\times$, and the result follows. □

Lemma 21. *If ℓ splits in $\mathfrak{O}_K$ as $\mathfrak{l}\bar{\mathfrak{l}}$ where $\mathfrak{l} = (\ell, a\omega + b)$, the map $\mathfrak{O}_K/\mathfrak{l} \to \mathbb{F}_\ell$ defined by $\omega \mapsto -b/a$ is a ring isomorphism with inverse $\mathbb{F}_\ell \to \mathfrak{O}_K/\mathfrak{l}$, $n \mapsto n$.*

Proof. We have $(a, \ell) = 1$. Consider the ring homomorphism

$$\pi : \mathbb{Z}[X] \longrightarrow \mathbb{F}_\ell,\ X \longmapsto -b/a.$$

Writing $\overline{a\omega + b} = c\omega + d$, we have $f(X) \equiv (aX + b)(cX + d) \pmod{\ell}$, hence $\pi(f(X)) = 0$ and so π induces a well-defined ring homomorphism

$$\pi' : \mathfrak{O}_K \longrightarrow \mathbb{F}_\ell,\ \omega \longmapsto -b/a\,.$$

Clearly, $\mathfrak{l} = (\ell, a\omega + b) \subseteq \ker(\pi')$, so $\mathfrak{O}_K/\mathfrak{l} \to \mathbb{F}_\ell,\ \omega \mapsto -b/a$ is well-defined. It is surjective, and the source and target both have size ℓ, so it is an isomorphism. □

Proposition 22. *If ℓ splits in $\mathfrak{O}_K$ as $\mathfrak{l}\bar{\mathfrak{l}}$ where $\mathfrak{l} = (\ell, a\omega+b)$, and $a\omega+b = c\overline{\omega}+d$, the map*

$$(\mathfrak{O}_K/\ell\mathfrak{O}_K)^\times/(\mathfrak{O}/\ell\mathfrak{O})^\times \longrightarrow \mathbb{F}_\ell^\times,\ x\omega + y \longmapsto (-xb/a + y)/(-xd/c + y)$$

is a group isomorphism.

Proof. It is the composition of the isomorphisms from Lemmata 20 and 21. □

Hence, we have an efficiently computable map $(\mathfrak{O}_K/\ell\mathfrak{O}_K)^\times/(\mathfrak{O}/\ell\mathfrak{O})^\times \to \mathbb{F}_\ell^\times$. Now if we start from an $\mathfrak{O}$-ideal $\mathfrak{a}$, we first find a generator α of $\mathfrak{a}\mathfrak{O}_K$, and map it to $\mathbb{F}_\ell^\times$ through the isomorphism. The choice of generator is non-canonical, and may lead to an ambiguity in the definition of this map: it actually is a map to a quotient $\mathbb{F}_\ell^\times/G$ where G is a subgroup of order $|\mathfrak{O}_K^\times/\mathfrak{O}^\times|$. In the case $K = \mathbb{Q}(\sqrt{-1})$, we have $G = \{\pm 1\}$.

In summary, to map an $\mathfrak{O}$-ideal $\mathfrak{a}$ to the group $\mathbb{F}_\ell^\times/G$:

1. Find α such that $\mathfrak{a}\mathfrak{O}_K = \alpha\mathfrak{O}_K$;
2. Write $\alpha/\overline{\alpha} = x\omega + y$ for $x, y \in \mathbb{Z}$;
3. Output $y - xb/a \bmod \ell$.

A.2 The inert case

When ℓ is inert in $\mathfrak{O}_K$, the polynomial $f(X)$ is irreducible in $\mathbb{F}_\ell$, so one can define $\mathbb{F}_{\ell^2} = \mathbb{F}_\ell[X]/f(X)$.

Lemma 23. *If ℓ is inert in $\mathfrak{O}_K$, the map*

$$\mathfrak{O}_K/\ell\mathfrak{O}_K \longrightarrow \mathbb{F}_{\ell^2},\ \omega \longmapsto X$$

is a field isomorphism.

Proof. The ring homomorphism $\mathbb{Z}[X] \to \mathbb{F}_{\ell^2},\ X \to X$ is surjective and has $f(X)$ and ℓ in its kernel, so induces a surjective homomorphism $\mathfrak{O}_K/\ell\mathfrak{O}_K \to \mathbb{F}_{\ell^2}$. The source and target are both fields of order ℓ^2, so it is an isomorphism. □

Proposition 24. *If ℓ is inert in $\mathfrak{O}_K$, the map*

$$(\mathfrak{O}_K/\ell\mathfrak{O}_K)^\times/(\mathfrak{O}/\ell\mathfrak{O})^\times \longrightarrow \mathbb{F}_{\ell^2}^\times/\mathbb{F}_\ell^\times,\ \omega \longmapsto X$$

is a group isomorphism.

Proof. The isomorphism from Lemma 23 maps the subfield $\mathfrak{O}/\ell\mathfrak{O} \subset \mathfrak{O}_K/\ell\mathfrak{O}_K$ to $\mathbb{F}_\ell \subset \mathbb{F}_{\ell^2}$. □

B Parameters

128-bit parameters. We choose $n_1 + n_2 = 37$ such that $f > 2^{256}$ as long as L_2 is sufficiently small. Taking $L_2 = 5$, we found

$$\begin{aligned}\alpha = &-600591808385180536757881465597002302416458558485126821764359031606300809784518\\ &+813882587493810077851957456371883857713360998173103581924791873490003540502291i\end{aligned}$$

with $f \approx 2^{259}$ and $f + 1$ being 2^{34}-smooth.

With the values given above, we choose to consider $L = L_1L_2L_3$ as the product of the 65 smallest primes split in $\mathfrak{O}_0$. We choose the prime characteristic to be $p = 4cL - 1$ for a cofactor $c = 335$ and it is of 528 bits. In that case, we ran SetUpCurve and found the curve $E\ y^2 = x(x^2 + Ax + 1)$ and the generators $P_E + R_E = (X_P : Z_P)$ and $Q_E = (X_Q : Z_Q)$ with

$$\begin{aligned}A = &\,609713185630972010670935559853144224720117201550156339729329469291386152543824 3\\ &\,055864040272032863712185595760942196122548954924073850778353532938364735822161 56i\\ &+615420305026332729418500746857711682502001687969362345023251556257118439903978 0\\ &\,742395058445175144114992421709686251794871460139566353871222520063101399614586 27\\ X_P = &\,649662766955987262712642653495434186656775206288198442731398218601213376093486 8\\ &\,205326740397761286604460870997720248971963308539550579872915224597495713887436 4i\\ &+305634628625616960782265852759568451995224409571691436301588240149740241903123 8\end{aligned}$$

$$
\begin{aligned}
&840670624444981744627571008631230840274776493077469214971783334692890435659953 89\\
Z_P = &189973908983857196032146564401173801303257676168549492806318181104067177233687 1\\
&896082178307025851721056136130174102800018891460928379654640783638913927324567 3i\\
+ &147536290345474169432241022277647529485993368751414461514247010168254478585270 1\\
&371735130053197252226479018559811817444901674957829523069202888563201609518015 1\\
X_Q = &246981879247557545763993087237034139044900367703281305381071612145231894926438 6\\
&939357853184333625837476984732641465403397791838400679320098122090073294669555 38i\\
+ &467369244782891495439499538266339319885587052010406129668243563967862726424360 8\\
&289416906246566645505718424801557253027871422018396695281554420753918776730390 43\\
Z_Q = &248137706349969050623203728715630356027266346888181717081745874717163517833887 9\\
&153729330447059068916808451099088106274980642029404219605481612432987265545559 47i\\
+ &538794422715739470837571010481586030525108148491049842442923442344220234238085 0\\
&721057015291475758024485148494701030191657429679617785705057885006956453741800 32
\end{aligned}
$$

over $\mathbb{F}_{p^2} = \mathbb{F}_p[i] = \mathbb{F}_p[X]/\langle X^2 + 1\rangle$.

256-*bit parameters.* We choose $n = 68$ and taking $L_2 = 5$ we obtained

$$
\begin{aligned}
\alpha = &-173278917128799924886584001437161562178110128043679327372340521 0\\
&526356347946603557614710265303485229586472132988844003836753101468\\
&59057269270267622717600758\\
+ &111067294716243081975130937217528372885477020011478178590825958748\\
&137054759443760832676453989566903528901601602870671704714448434265\\
&277819658117244388003679i.
\end{aligned}
$$

In this case, $f \approx 2^{516}$ and $f + 1$ is 2^{74}-smooth.

With the values given above, we choose to consider $L = L_1L_2L_3$ as the product of the 75 smallest primes split in $\mathfrak{O}_0$. We choose the prime characteristic to be $p = 4cL - 1$ for a cofactor $c = 256$ and it is of 625 bits. In that case, we ran SetUpCurve and found the curve $E\ y^2 = x(x^2 + Ax + 1)$ and the generators $P_E + R_e = (X_P : Z_P)$ and $Q_E = (X_Q : Z_Q)$ with

$$
\begin{aligned}
A = &147275998382645776665008425032549727015261439838745102087356793299\\
&460220756816220368842384577213216019900037345726944538937136153946 5\\
&703279016172012023515250943437740735287615007324412236 2i\\
+ &119077054972255322960390267599689318914164103457996137020065802336\\
&870546047625290680884205260060412822450168448166207622464110984053 4\\
&550710474887150004712347560412671107331386487493432492 4\\
X_P = &492555444431645203474344564742525793139043136885801237701636936324\\
&951232107235840812476951484114375000757820280578682960345188158124 6\\
&711790201217844297691962656275151248012386830047563858 2i
\end{aligned}
$$

$$
\begin{aligned}
&+ 160728320760902619108152017541647965397628920335381690975773174016\\
&\quad 134537384151443265880313548095928760024020600025927579727425391629 6\\
&\quad 964857126492874282292041837267634867777678867995580993 7\\
Z_P = &\ 508856872348785710159152814624363239736852521142590779448941887712\\
&\quad 009376737000234255613263701859290137308144301018101329613150798683 1\\
&\quad 761382654380340989093350777749769785210774964944914007 2i\\
&+ 652483897787328979568655540089891443070986122621973121391553673615\\
&\quad 183320693073281446314846902676711554863889174101207956167904447784 2\\
&\quad 002701136720971059427771090112298120391404787559952314 4\\
X_Q = &\ 828879044525035006483969914772350545504678499643470945911426330145 1\\
&\quad 135553669945348939150398280114042420963163094021928448777402648074 37\\
&\quad 682905326644349765739826877381563633921892168505886 23i\\
&+ 384115325031866432970792717602295322333784629575954041931889404422 7\\
&\quad 418737588769271073784057747785462633156807245225589715656676533190 01\\
&\quad 885882149667396804680202719262505433592686912036183 67\\
Z_Q = &\ 763250664170994807123138778288169595703675540573944888418539630497 3\\
&\quad 670285126096989935224247985278875178529528896965522748454890953230 07\\
&\quad 554634675661332290610653900770818966878567941589328 92i\\
&+ 704147114673890304149013239624126870080211320013996633286905168262 6\\
&\quad 261316790162896836150288154800439070365635435825289417775133366481 79\\
&\quad 684495850964638523995005358325119476222411080717854 64
\end{aligned}
$$

over $\mathbb{F}_{p^2} = \mathbb{F}_p[i] = \mathbb{F}_p[X]/\langle X^2 + 1\rangle$.

Note that we were far from exhausting the search spaces in either case and it may be possible to find smoother solutions and, consequently, to further accelerate the setup.