A Curved Path to Post-Quantum: Cryptanalysis and Design of Isogeny-based Cryptography

Simon-Philipp Merz

Information Security Group
Royal Holloway, University of London

This dissertation is submitted for the degree of
Doctor of Philosophy

April 2023

Declaration

These doctoral studies were conducted under the supervision of Prof. Simon R. Blackburn and Prof. Christophe Petit.

The work presented in this thesis is the result of original research carried out by myself, in collaboration with others, whilst enrolled in the Information Security Group as a candidate for the degree of Doctor of Philosophy. This work has not been submitted for any other degree or award in any other university or educational establishment.

Simon-Philipp Merz
April 2023

Acknowledgements

This thesis is one of the products of studying and doing research in isogeny-based cryptography during the last four years. Many people have contributed in some way to making this not only a productive and interesting but also a very delightful, enjoyable and adventurous time.

First and foremost, I would like to thank my supervisors Simon Blackburn and Christophe Petit. Thank you, Simon, for taking me on as a PhD student, for regular meetings and helpful advice, and for often ending a meeting with a comment that left me with a smile for the rest of the day. Thank you, Christophe, for first introducing me to the field of cryptography during my master's degree and for including me in your research group despite administrative issues, for sharing your research ideas that have led to some papers included in this thesis and for sharing your contacts to enable other collaborations.

I am grateful to all of my coauthors for their effort and work during our collaborations. A particular thanks to Péter Kutas for patiently answering many of my questions, discussing new projects and becoming a friend over the duration of the PhD. Thank you to everyone in the isogeny reading group for studying recent papers together, in particular to Boris Fouotsa and Andrea Basso for interesting discussions during and after the reading groups. Further, thanks to all the friends I have made and the acquaintances which I had the pleasure to meet at various conferences and workshops.

I would like to thank Luca De Feo for hosting me as an intern at IBM Research Zürich during the last summer of my studies. Thanks to Luca and Ward for interesting research discussions and thanks to all of the members of the Foundations of Cryptography group for the welcoming atmosphere, interesting conversations and superb leisure activities.

Thank you to Wouter Castryck and Frederik Vercauteren for inviting me to Leuven for a research visit. The stimulating work and the very warm welcome by all the members of COSIC made the time fly by very quickly.

Thanks to Chloe Martindale and James McKee for agreeing to be my examiners and for reading this document in detail. I thank Claire Hudson for being the most reliable point of contact to guide me through the university's administrative jungle. I would like to thank Martin Albrecht for acting as my advisor in the annual reviews, for keeping different cryptography reading groups alive and for a pint or two at the Crown.

Thanks to my friends from Royal Holloway – Alpesh Bhudia, Ben Phillips and Erin Hales, Balázs Mezei, Eamonn Postlethwaite, Fernando Virdia, Jeroen Pijnenburg, Jodie Knapp, Joe Rowell, Liam Medley, Rob Markiewicz – for great conversations during countless tea breaks, for nice wedding parties, for our running competitions, for many dinners and movie nights, for the Card Club and Egham hill bets, for various trips before the pandemic, for making the lockdowns more bearable and for our sociable gatherings in many different places.

Thank you to all of the fellow students who accompanied my mathematical development over the years. In particular, thanks to Lelia Hanslik from my undergraduate studies for keeping me up to date with the gossip of our Berlin student group after I left, whenever I needed it; thanks to Alexander Schell for strolls with engaging conversations about much more than maths that made time always pass so quickly; thanks to Sivert Aesnæss, Hector Papoulias and Alec Letcher for sharing so much of your enthusiasm with me, spending holidays together and inviting me to your homes all over Europe; and thanks to Wieland Goetzke for enjoying many Burger Mondays at the Rickety Press with me, treating me with home-cooked food and having a place for me to stay at New College whenever I came back to Oxford.

I would like to thank my brother, Julian, for many discussions that had absolutely nothing to do with my work and the regular reminders that it is always a good time for an adventure, and my parents for being supportive from the start and always trusting me with my life choices.

Thanks to everyone not mentioned here, but who, in some way, contributed actively or passively to the PhD journey.

Finally, thank you to Lenka for her affection, relentless support and everything else!

Publications

The content of this thesis is based on the following publications:

1. Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Charlotte Weitkämper. One-way functions and malleability oracles: Hidden shift attacks on isogeny-based protocols. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 242–271. Springer, Heidelberg, October 2021.

2. Simon-Philipp Merz, Romy Minko, and Christophe Petit. Another look at some isogeny hardness assumptions. In Stanislaw Jarecki, editor, *CT-RSA 2020*, volume 12006 of *LNCS*, pages 496–511. Springer, Heidelberg, February 2020.

3. Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Antonio Sanso. Cryptanalysis of an oblivious PRF from supersingular isogenies. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part I*, volume 13090 of *LNCS*, pages 160–184. Springer, Heidelberg, December 2021.

4. Tako Boris Fouotsa, Péter Kutas, Simon-Philipp Merz, and Yan Bo Ti. On the isogeny problem with torsion point information. In Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe, editors, *PKC 2022, Part I*, volume 13177 of *LNCS*, pages 142–161. Springer, Heidelberg, March 2022.

5. Luca De Feo, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Simon-Philipp Merz, Lorenz Panny, and Benjamin Wesolowski. SCALLOP: Scaling the CSI-FiSh. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 345–375. Springer, Heidelberg, May 2023.

The following works were also written (or completed) during the author's PhD studies at Royal Holloway, University of London:

6. Simon-Philipp Merz and Christophe Petit. Factoring products of braids via garside normal form. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 646–678. Springer, Heidelberg, April 2019.

7. Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Charlotte Weitkämper. On adaptive attacks against Jao-Urbanik's isogeny-based protocol. In *Progress in Cryptology-AFRICACRYPT 2020: 12th International Conference on Cryptology in Africa, Cairo, Egypt, July 20–22, 2020, Proceedings 12*, pages 195–213. Springer, 2020.

8. Steven D. Galbraith, Robert Granger, Simon-Philipp Merz, and Christophe Petit. On index calculus algorithms for subfield curves. In Orr Dunkelman, Michael J. Jacobson Jr., and Colin O'Flynn, editors, *SAC 2020*, volume 12804 of *LNCS*, pages 115–138. Springer, Heidelberg, October 2020.

9. Jeremy Booher, Ross Bowden, Javad Doliskani, Tako Boris Fouotsa, Steven D. Galbraith, Sabrina Kunzweiler, Simon-Philipp Merz, Christophe Petit, Benjamin Smith, Katherine E. Stange, Yan Bo Ti, Christelle Vincent, José Felipe Voloch, Charlotte Weitkämper, and Lukas Zobernig. Failing to hash into supersingular isogeny graphs. Cryptology ePrint Archive, Report 2022/518, 2022. https://eprint.iacr.org/2022/518.

10. Wouter Castryck, Marc Houben, Simon-Philipp Merz, Marzio Mula, Sam van Buuren, and Frederik Vercauteren. Weak instances of class group action based cryptography via self-pairings. To appear at CRYPTO 2023. Preprint available at https://eprint.iacr.org/2023/549, 2023

All of this research was supported by the EPSRC and the UK government as part of the Centre for Doctoral Training in Cyber Security at Royal Holloway, University of London (Grant EP/P009301/1).

Abstract

This thesis presents multiple results concerning the cryptanalysis and design of isogeny-based primitives and advanced protocols which aim to provide security in the presence of classical and quantum adversaries. The problems underlying these cryptographic protocols often provide additional information to the adversary such as the degree of a secret isogeny or the evaluation of the isogeny on given points. We study the impact of this additional information on the security of protocols at different levels, spanning from foundational key agreement to more advanced cryptographic constructions with real-world applications.

First, we consider different attacks on the foundational problem underlying (variants of) the Supersingular Isogeny Diffie–Hellman key exchange (SIDH), which reveals torsion point images under a secret isogeny. We extend previous torsion point attacks to slightly less imbalanced parameter sets and we present a reduction of the problem underlying certain overstretched and imbalanced SIDH variants to an abelian hidden shift problem that can be solved in subexponential time on a quantum computer. We briefly summarise the idea of a recent series of devastating attacks on SIDH due to Castryck and Decru [CD22], Maino and Martindale [MM22], and Robert [Rob22a].

Second, we cryptanalyse multiple isogeny-based hardness assumptions used in the security proofs of more advanced cryptographic constructions such as undeniable signatures by Jao and Soukharev and an OPRF by Boneh, Kogan and Woo. We provide efficient attacks against the hardness assumptions and we show how the attacks extend to breaking the advanced cryptographic protocols themselves. For the suggested parameters, the attacks allow us to forge undeniable signatures and break the pseudorandomness of the OPRF. These attacks predate the aforementioned papers attacking SIDH.

Third, we provide an efficient algorithm to compute a secret isogeny of a specific degree between supersingular elliptic curves given their endomorphism rings and some torsion point images under the secret isogeny. This reduction of the problem underlying SIDH-like protocols to the problem of computing endomorphism rings provides a new lower bound on the size of the finite field the supersingular curves need to be defined over.

Finally, we present a new group action of an imaginary quadratic order's class group on the set of oriented supersingular curves, such that the class group structure is easily computable. This data is required to uniquely represent and efficiently act by arbitrary group elements, which is necessary for example in the CSI-FiSh signature scheme. The index-calculus algorithm used in CSI-FiSh to compute the class group structure of the acting class group in CSIDH-512 rules out much larger parameters, a limitation that is particularly problematic in light of the ongoing debate regarding the quantum security of cryptographic group actions. A careful choice of parameters allows us to instantiate our group action in practice for a security level equivalent to CSIDH-1024, a security level currently out of reach using the index-calculus-based methods.

Table of contents

CHAPTER 1

Introduction

Cryptography is the research of techniques for secure communication and storage in the presence of adversaries, allowing to reduce trust in third parties. For instance, securing communication takes the form of guaranteeing the confidentiality, integrity or authenticity of messages.

The invention of methods to keep written messages out of the view of prying eyes can be traced almost as far back as writing itself. Early examples of cryptography include ciphers that rearrange or substitute letters systematically. In the 20th century, cryptography as a discipline, which until then was often thought of as a linguistic exercise, started changing radically. The development of rotor cipher machines and the revolutionary arrival of computers increased the complexity of cryptographic methods and placed the discipline somewhere at the intersection of mathematics, computer science and electrical engineering.

Another groundbreaking revolution happened to cryptography in 1976, when Diffie and Hellman introduced *public key cryptography* [DH76]. Public key cryptography allows multiple parties to establish secure communication over insecure channels without prior agreement on shared key material. Nowadays, public key cryptography is used by billions of people on a daily basis, enabling for instance modern digital communication and crucial services of contemporary society such as electronic payment systems.

The security of public key cryptographic constructions in turn relies on the hardness of certain computational mathematical problems. The most prominent systems currently deployed are based on the hardness of the computational problems of factoring large integers and computing discrete logarithms (either over finite fields or on elliptic curves defined over finite fields). Unfortunately, there is no proof that these computational problems are actually hard to solve. Our belief that these problems are indeed hard relies entirely on several decades of cryptanalytic efforts by experts from computational mathematics that did not lead to fundamental progress in solving these problems efficiently on classical computers processing binary information.

However, when taking into account machines with a different computational model, this may no longer be true. In fact, using (variants of) an algorithm due to Shor [Sho94] and access to a *quantum computer* with sufficient processing power and error correction,

the mathematical problems underlying the cryptosystems widely deployed today (such as the factorisation of integers, or discrete logarithm problems) can be solved efficiently.

Independent of the views on the plausibility or the concrete timeline of the development of large-scale quantum computers, the danger of "harvest now, decrypt later" attacks as well as the long time it takes to develop and deploy new cryptographic standards means that it is crucial to understand quantum-secure cryptography now. *Post-quantum cryptography* is the subject area which is concerned with cryptographic algorithms that remain secure in the presence of quantum computers. To this end, many new computational problems are being proposed and protocols based on these new problems are suggested for real-world use.

To further encourage research in post-quantum cryptography and to standardise the first promising candidates, the National Institute of Standards and Technology (NIST) launched a standardisation process in 2016 [NIS16]. While this process concluded in 2022 and the winning proposals were put forward for standardisation, the area of post-quantum cryptography has greatly advanced in recent years and another standardisation process for quantum-resistant digital signatures is going to start in 2023 [NIS22].

The work exhibited in this thesis contributes to the analysis and further development of *isogeny-based cryptography*, one branch of post-quantum cryptography. Isogenies are non-constant rational maps between elliptic curves that are also group homomorphisms. For readers not familiar with this terminology we introduce it in more detail in Section 2.2. The central hard problem underlying isogeny-based cryptography is to compute an isogeny between two given elliptic curves, when it exists. This can be seen as a natural analogue to the classical discrete logarithm problem on elliptic curves, as scalar multiplication corresponds to an isogeny from one curve to itself. Further, isogenies were used for cryptanalysis of the classical discrete logarithm problem on elliptic curves [GHS02]. As such, cryptographers have been familiar with some aspects of isogeny-based cryptography for a while.

In 2006, Stolbunov and Rostovtsev were the first to see the potential for a quantum-secure key exchange using a group action on a set of ordinary elliptic curves computed by the means of isogenies [RS06]. A very similar construction had previously been presented by Couveignes [Cou06]. Yet, without noticing the selling point of quantum-resistance, the rather inefficient construction, which did not seem to offer any practical advantages over other competitors in classical cryptography, was not published until after the work by Stolbunov and Rostovtsev.

The problem underlying the Couveignes–Rostovtsev–Stolbunov (CRS) key exchange was originally conjectured to take exponential time to solve on a quantum computer.

However, Childs, Jao and Soukharev showed that it reduces to a hidden shift problem [CJS14], which can be solved in quantum subexponential time using an algorithm due to Kuperberg [Kup05], further adding to the inefficiency of CRS.

Supersingular elliptic curves were first used in cryptography in a proposal for hash functions [CLG09]. A few years later, Jao and De Feo constructed the supersingular isogeny Diffie–Hellman (SIDH) key exchange [JD11]. Trying to avoid the commutative structure giving rise to the quantum subexponential attack by Childs–Jao–Soukharev on the CRS key exchange, the SIDH key exchange takes place in the much less structured full supersingular isogeny graph. To complete the key exchange despite the lack of commutative structure, Jao and De Feo suggested to send some auxiliary information with the supersingular elliptic curves in the key exchange. However, this additional information enabled active attacks on SIDH [GPST16] and for imbalanced parameters was shown to lead to a break of the problem underlying SIDH [Pet17, dQKL+21]. Multiple chapters of this thesis are concerned with improvements of these attacks and new reductions between different computational problems in isogeny-based cryptography in the presence of the additional information.

The only isogeny-based submission to NIST's post-quantum standardisation process, SIKE [JAC+17], which is a variant of SIDH using balanced parameters, made it all the way to the final round of the process, convincing the community with its very compact keys and reasonable speed. Unfortunately, an attack due to Castryck and Decru exploiting SIDH's auxiliary information broke SIKE efficiently in a spectacular way [CD22]. Related work followed soon after, showing that there is little hope to salvage SIDH in its most efficient form [MM22, Rob22a]. Countermeasures against the attacks were proposed, however they lead to larger keys and a slower key exchange [Mor22, Fou22].

Building upon several ideas from an improvement to CRS by De Feo, Kieffer and Smith [DKS18], a cryptographic group action of a class group of an imaginary quadratic order on the set of *supersingular* elliptic curves defined over a prime field was introduced in [CLM+18]. This group action is at the base of the scheme CSIDH, which seems unaffected by the SIDH attacks, keeps being improved and gives rise to an efficient non-interactive key exchange. Further schemes such as SQISign [DKL+20], a very compact isogeny-based signature scheme, and a new isogeny-based group action called SCALLOP [DFK+23a], which we will introduce in Chapter 6 of this thesis, are both based on new hardness assumptions. Additionally, there are plentiful new applications built on top of isogeny-based primitives. All of this shows that the area is still gaining more traction. Thus, the break of SIDH was not the end of isogeny-based cryptography, but just the start of another chapter.

Outline of the thesis

In this thesis, we present research carried out as part of multiple publications in collaboration with other authors. We present multiple attacks on SIDH variants, cryptanalyse advanced protocols based on SIDH variants, provide a new reduction between isogeny-based problems, and finally introduce a new isogeny-based group action.

In the following, we give a brief description of each chapter of the rest of this thesis.

Chapter 2 provides the mathematical background of isogeny-based cryptography necessary to follow the technical contributions of the subsequent chapters. Moreover, it briefly introduces the main isogeny-based key exchange protocols, and presents some computational hardness assumptions underlying their security. We point towards further work for interested readers.

Chapter 3 surveys multiple attacks on balanced and imbalanced SIDH variants using the provided torsion point information. We briefly recall so-called *torsion point attacks* from a line of work started by Petit [Pet17]. We present an unpublished improvement to these ideas reducing the imbalance of SIDH parameters required for the attacks to work. Next, we include a reduction of the problem underlying imbalanced SIDH to a hidden shift problem. This reduction was previously published as

Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Charlotte Weitkämper. One-way functions and malleability oracles: Hidden shift attacks on isogeny-based protocols. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 242–271. Springer, Heidelberg, October 2021.

For the sake of completeness, we sketch the idea behind the recent efficient attacks on (balanced) SIDH due to Castryck and Decru [CD22] and discuss the impact of related work that followed shortly after, e.g. a more direct attack which uses tools from our work presented in Chapter 5 and a generalisation to general starting curves [MM22, Rob22a].

The author of this thesis contributed to all aspects of the improved torsion point attacks and the reduction of imbalanced SIDH to a hidden shift problem.

Chapter 4 discusses multiple hardness assumptions underlying advanced protocols built from the SIDH key exchange. More precisely, the assumptions were introduced in the context of SIDH-based undeniable signatures and oblivious pseudorandom functions. First, we present an efficient attack on the hardness assumptions underlying an undeniable signature by Jao and Soukharev [JS14]. Further, we describe how our attack extends

to an (exponential time) attack on the undeniable signature scheme itself, breaking the recommended parameter sets. This cryptanalysis was previously published as

Simon-Philipp Merz, Romy Minko, and Christophe Petit. Another look at some isogeny hardness assumptions. In Stanislaw Jarecki, editor, *CT-RSA 2020*, volume 12006 of *LNCS*, pages 496–511. Springer, Heidelberg, February 2020.

Further, we cryptanalyse an oblivious pseudorandom function (OPRF) proposed by Boneh, Kogan and Woo [BKW20]. While parameters for the OPRF were chosen to defend against our previously mentioned attack on undeniable signatures, we provide new attacks on the so-called *auxiliary one-more assumption* introduced by Boneh, Kogan and Woo to argue the security of their SIDH-based OPRF. We propose multiple attacks that not only break the assumption but also the pseudorandomness of the OPRF. First, we give a polynomial-time attack for which we present a simple countermeasure that can be included in the OPRF protocol. Second, we present a subexponential attack that succeeds in the presence of this countermeasure. The attacks break all security parameters suggested by Boneh, Kogan and Woo. Finally, we show that one of the OPRF parameters needs to be generated using a trusted third party to avoid further attacks. This work was previously published as

Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Antonio Sanso. Cryptanalysis of an oblivious PRF from supersingular isogenies. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part I*, volume 13090 of *LNCS*, pages 160–184. Springer, Heidelberg, December 2021.

The author of this thesis contributed to all aspects of both of the papers.

Chapter 5 presents a new reduction from the problem of recovering isogenies of a specific degree between two supersingular elliptic curves to the problem of computing their endomorphism rings, assuming certain torsion point images under the sought isogeny are provided. The reduction applies for balanced parameters and for arbitrary fixed degrees. This work was previously published as

Tako Boris Fouotsa, Péter Kutas, Simon-Philipp Merz, and Yan Bo Ti. On the isogeny problem with torsion point information. In Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe, editors, *PKC 2022, Part I*, volume 13177 of *LNCS*, pages 142–161. Springer, Heidelberg, March 2022.

One consequence of this chapter is a new lower bound for the size of the finite field used to define the curves in SIDH-like schemes such as B-SIDH. Further, we introduced a subroutine that allows us to evaluate non-smooth degree isogenies if the endomorphism

rings of the domain and codomain are known. This has found application for example to generalise recent direct attacks on SIDH [Wes22c]. Compared to the published version mentioned above, the chapter explains how the reduction for SIDH-like schemes still applies in the presence of countermeasures against the recent SIDH attacks such as masking the degree or masking the torsion point images by multiplication with a scalar.

The author of this thesis contributed to all aspects of the paper.

Chapter 6 proposes a new isogeny-based group action called SCALLOP. Similar to CSIDH, it is the action of the class group of an imaginary quadratic order on a set of supersingular elliptic curves. In SCALLOP, the quadratic order used has a large prime conductor inside an imaginary quadratic field of small discriminant. In this case, one has easy formulas to compute the structure of the class group. For a certain type of signature schemes this data is required as it is pivotal to uniquely represent, and efficiently act by arbitrary group elements of the class group. This chapter is for all practical purposes identical to the paper previously published as

Luca De Feo, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Simon-Philipp Merz, Lorenz Panny, and Benjamin Wesolowski. SCALLOP: Scaling the CSI-FiSh. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 345–375. Springer, Heidelberg, May 2023.

The author of this thesis contributed primarily to the parameter generation for the SCALLOP group action.

CHAPTER 2

Preliminaries

In this chapter, we introduce the necessary mathematical background to enable the reader to follow the rest of this thesis. Our account should be sufficient for this purpose, though it only scratches the surface of the rich mathematical theory underlying and relating to isogeny-based cryptography. For a more complete introduction and to get a better understanding of the objects treated in this work, we refer to Silverman [Sil09] on the topic of elliptic curves and to Voight [Voi21] for quaternion algebras. For basic notions such as morphisms and varieties, we refer to any introductory textbook on algebraic geometry, e.g. [Sha94]. For a brief introduction to isogeny-based cryptography, we recommend De Feo's article [DF17] or the more computational introduction to SIDH by Costello [Cos19].

Apart from the mathematical background, we will use this chapter to fix some notation, recall the core ideas underlying the most prominent isogeny-based key exchange protocols and survey some of the hardness assumptions used in isogeny-based cryptography.

2.1 Notation and terminology

Below, we provide a list of notation and terminology that will be used without further explanation throughout the thesis.

- For a field k, the *algebraic closure* of k is denoted by $\overline{k}$.
- For a field k, we denote the *projective space* of dimension n by $\mathbb{P}^n(k)$, which is the quotient set of $k^{n+1}\backslash 0$ by the equivalence relation of componentwise scalar multiplication.
- By $\left(\frac{n}{p}\right)$ we denote the *Legendre symbol*, equal to 0 if p divides n, 1 if n is a non-zero square modulo p, and -1 otherwise.
- Let f and g be functions $\mathbb{N} \to \mathbb{R}$. We use the standard Landau notation $f \in O(g)$ to mean that f grows at most as fast as g, i.e. $\limsup_{n\to\infty} \left|\frac{f(n)}{g(n)}\right| < \infty$.
- We write $O(\text{poly}(x))$ for quantities asymptotically upper bounded by a polynomial in x. Sometimes, we may want to omit factors polynomial in $\log p$, where p is the characteristic of the finite field we are working with. In this case, we will abbreviate $O(g \cdot \text{poly}(\log(p)))$ by $O^*(g)$. Polynomial time without explicitly mentioning the variables means "polynomial in the representation size of the input".
- An algorithm is called *efficient* if the execution time is in $O(\text{poly}(\lambda))$, where λ denotes the security parameter of the underlying cryptographic scheme.
- We call a function $f : \mathbb{N} \to \mathbb{R}$ *negligible*, if $f \in O(x^{-c})$ for every positive integer c. For probabilistic algorithms, we say that something happens with *overwhelming probability* if the probability of its complement is given by a negligible function of the input length.
- A *B-smooth* integer n only has prime factors smaller than B, where B is called the *smoothness bound*. We sometimes say that n is *smooth*, meaning that the smoothness bound B of n is in $O(\text{poly}(\log(n)))$.
- We call an integer *B-powersmooth* when all its prime power divisors are smaller than B. As for smooth numbers, we may refer to an integer n as *powersmooth* when $B \in O(\text{poly}(\log(n)))$.
- We call a function $f : \{0,1\}^* \to \{0,1\}^*$ *one-way* if f can be computed efficiently, but any polynomial-time randomised algorithm trying to compute a preimage,

i.e. an element in the domain that evaluates to a given element in the codomain, succeeds with negligible probability.

- Given any function, by having *oracle access* to this function we mean that it is feasible to evaluate the function at any possible element efficiently. We assume that the oracle acts like a black box such that one query with an element from the domain outputs the corresponding value of the function.
- log refers to the logarithm in base 2.

2.2 Mathematical background

First, we introduce the mathematical preliminaries necessary to describe the main isogeny-based primitives.

2.2.1 Elliptic curves

Elliptic curves have been studied in their full generality in mathematics for many years. Since the start of the 21st century, they played a ubiquitous role in cryptography as they could be used to build faster and more compact cryptosystems. After defining elliptic curves in general, we will move towards the more special cases that are relevant for isogeny-based cryptography.

Definition 2.2.1. An elliptic curve over a field k is a pair $(E, \mathcal{O}_E)$, where E is a smooth projective curve over k of genus 1 and $\mathcal{O}_E$ a distinguished base point on E defined over k.

This means an elliptic curve is a projective variety and all of its $\overline{k}$-rational points are given by the solutions over k to a homogeneous equation in three variables. Throughout the thesis, we will denote the points of an elliptic curve E over some extension k^{ext} by $E(k^{\text{ext}})$. It turns out that the equations defining elliptic curves can be transformed to be of a very specific form. Different families of polynomials of such a specific form describing elliptic curves are called *elliptic curve models*.

Elliptic curve models. For simplicity, we will henceforth assume that the field k has characteristic different from 2 or 3. This assumption ensures that every elliptic curve is birationally equivalent to an elliptic curve in the short Weierstraß model given in the following definition. For the general equation and the transformations to get the simplified Weierstraß form, we refer to [Sil09, III].

Definition 2.2.2. The *Weierstraß model* of an elliptic curve E over a field k is given by an equation of the form

$$Y^2Z = X^3 + aXZ^2 + bZ^3,$$

with $a, b \in k$ and $4a^3 + 27b^2 \neq 0$, with the base point at $(0 : 1 : 0)$.

When the field of definition is not implicit, we may write E/k to mean the curve E over the field k. The point $(0 : 1 : 0)$, denoted by $\mathcal{O}_E$ and called the *point at infinity*, is the only point with $Z = 0$ on the curve. Rewriting the coordinates with $x := X/Z$ and $y := Y/Z$, we can equivalently consider the affine form of the Weierstraß equation

$$y^2 = x^3 + ax + b \tag{2.1}$$

with the additional point at infinity. Since every elliptic curve over k with $\mathrm{char}(k) \neq 2, 3$ can be written in the Weierstraß model [Sil09, III], it is often used as a canonical way of representing elliptic curves or to prove theoretical results.

For the purpose of exposition, we will stick to the Weierstraß model. However, for practical implementations of cryptographic algorithms discussed in this thesis, usually the *Montgomery* or *Edwards* models of elliptic curves are used. While not every curve admits a rational Montgomery or Edwards model, the arithmetic on elliptic curves can be executed very efficiently on these curves. In particular, Montgomery curves come with efficient x-only arithmetic and the celebrated *Montgomery ladder* for fast scalar multiplication [Mon87]. A helpful overview on the efficient algorithms for Montgomery curves is given in [CS18]. Similar practical advantages come with curves in the Edwards model and they have complete addition formulas, meaning the same formula can be used for point doubling and point addition [BL07].

The group law. Elliptic curves have been interesting in cryptography and, much earlier, in pure mathematics because points on an elliptic curve form an abelian group. This mathematical structure can be visualised by the so-called *chord and tangent rule*: By Bézout's theorem, we know that an elliptic curve, defined by a cubic equation, and any line in $\mathbb{P}^2$ intersect in exactly three points, counting with multiplicities. The group law can be defined by requiring any three co-linear points to sum to the neutral element given by the point at infinity $\mathcal{O}_E$. (See [Sil09, III.2] for a proof that this indeed defines a group.) The group law is depicted in Fig. 2.1 with all vertical lines converging at $\mathcal{O}_E$.

By adding points to themselves, the group law gives rise to scalar multiplication.

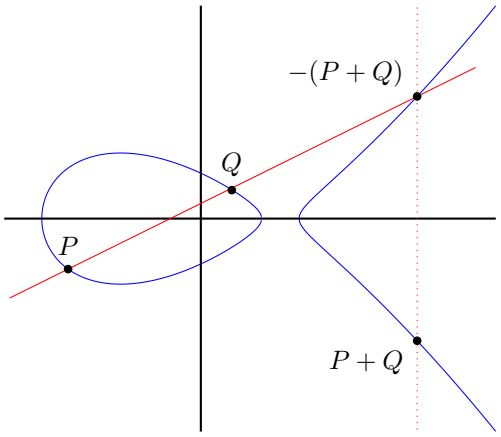

Fig. 2.1 Group law on the elliptic curve $y^2 = x^3 - 2x + 1$ defined over $\mathbb{R}$

Definition 2.2.3. For any $n \in \mathbb{Z}$, we write $[n] : E \to E$ for the *scalar multiplication by* n on an elliptic curve E/k, which adds n copies of a point together. The kernel of this map considered over $\overline{k}$ is called the *n-torsion subgroup* of E, denoted by $E[n]$.

The following proposition describes the structure of $E[n]$.

Proposition 2.2.4. *Let E/k be an elliptic curve and $n \in \mathbb{Z}$.*

- *$E[n] \cong \mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}$, if the characteristic of k does not divide n*
- *If k is of characteristic $p > 0$, then*

$$E[p^i] \cong \begin{cases} \mathbb{Z}/p^i\mathbb{Z} & \text{for any } i \geq 0, \text{ or} \\ \mathcal{O}_E & \text{for any } i \geq 0 \end{cases} \tag{2.2}$$

Proof. [Sil09, III, Cor. 6.4]. □

The behaviour of an elliptic curve on the p^i-torsion impacts the general structural properties of the curve, justifying a name for the two alternatives above.

Definition 2.2.5. An elliptic curve E/k is called *supersingular* if $\mathrm{char}(k) = p > 0$ and $E[p^r] \cong \mathcal{O}_E$ for one/all $r \in \mathbb{Z}_{>0}$. Otherwise, we say E is *ordinary*.

There are alternative characterisations of supersingular elliptic curves, see for instance [Sil09, V, Thm. 3.1] and Proposition 2.2.9.

Isomorphisms of elliptic curves. Two elliptic curves E, E' are said to be *isomorphic* over a field k if there exists a rational isomorphism whose coefficients are defined over k from one curve to the other. Clearly, this creates an equivalence relation on the set of elliptic curves over k. In this thesis, we will usually be interested in isomorphism classes of elliptic curves over $\overline{k}$. Isomorphisms preserving the Weierstraß form of an equation are linear changes of coordinates, and looking at Eq. (2.1) allows us to verify that the only such maps are

$$(x, y) \mapsto (u^2 x', u^3 y') \tag{2.3}$$

for some $u \in \overline{k}$. Note that linear transformations preserve the co-linearity of points and thus the map in Eq. (2.3) preserves the group law. The map defines an isomorphism between the two elliptic curves with Weierstraß equations $y^2 = x^3 + au^4 x + bu^6$ and $(y')^2 = (x')^3 + ax' + b$ respectively. These isomorphism classes of elliptic curves over $\overline{k}$ are encoded by the following invariant.

Definition 2.2.6. Let E/k be an elliptic curve given in its Weierstraß model by the affine equation $y^2 = x^3 + ax + b$. The *j-invariant* is defined as

$$j(E) := 1728 \frac{4a^3}{4a^3 + 27b^2}.$$

Considering the transformations given by Eq. (2.3) and noting that they are are isomorphisms between elliptic curves with Weierstraß equations $y^2 = x^3 + au^4 x + bu^6$ and $(y')^2 = (x')^3 + ax' + b$ one sees that the j-invariant is indeed invariant under these transformations.

Proposition 2.2.7 ([Sil09, III.1, Prop. 1.4])**.** *Two elliptic curves E/k and E'/k are isomorphic over $\overline{k}$ if and only if they have the same j-invariant.*

A *twist* between curves defined over k is an isomorphism that is defined over the algebraic closure of k but not over k itself. For a finite field k, choosing u in Eq. (2.3) such that $u \notin k$ and $u^2 \in k$ shows that every curve over k has a quadratic twist.

The hard problem underlying classical elliptic curve based cryptography is the problem of computing *discrete logarithms* for carefully chosen elliptic curves over finite fields. That is, given two points $P, Q := [n]P$ on some elliptic curve, the task is to recover the integer n. Assuming P is chosen such that $\langle P \rangle$ contains most of the curve's points, the number of points on such an elliptic curve is one important parameter when estimating the hardness of this problem.

Theorem 2.2.8 (Hasse bound)**.** *Let $E/\mathbb{F}_q$ be an elliptic curve, then*

$$\#E(\mathbb{F}_q) = q + 1 - t \quad \text{with} \quad |t| \leq 2\sqrt{q}.$$

The integer t in Theorem 2.2.8 is called the *trace* (of Frobenius) of the curve $E/\mathbb{F}_q$. The trace, and thus the precise number of points, can be computed efficiently using Schoof's algorithm [Sch85]. By the following result, this allows us to decide efficiently whether a curve is supersingular or not.

Proposition 2.2.9 ([Was08, Prop. 4.31])**.** *Let $E/\mathbb{F}_q$ be an elliptic curve, where q is the power of a prime p. Then $\#E(\mathbb{F}_q) = q + 1 - t$ and E is supersingular if and only if $t \equiv 0 \bmod p$, i.e. E is supersingular if $\#E(\mathbb{F}_q) \equiv 1 \bmod p$. In particular, if $q = p \geq 5$ then $\#E(\mathbb{F}_p) = p + 1$.*

By choosing a certain prime field, we can thus control the number of points on a supersingular elliptic curve. This is one of the reasons why they are used in isogeny-based cryptography. Further practical considerations are that isomorphism classes of supersingular elliptic curves can be efficiently represented and the existence of the following explicit formula to compute the number of such isomorphism classes.

Proposition 2.2.10 ([Sil09, V.Thm. 4.1])**.** *Let $\mathcal{S}(p)$ denote the set of j-invariants of supersingular elliptic curves defined over a field of characteristic $p \geq 5$. Then $\mathcal{S}(p) \subset \mathbb{F}_{p^2}$ and*

$$\#\mathcal{S}(p) = \left\lfloor \frac{p}{12} \right\rfloor + \begin{cases} 0, & \text{if } p \equiv 1 \bmod 12 \\ 1, & \text{if } p \equiv 5 \text{ or } 7 \bmod 12 \\ 2, & \text{if } p \equiv 11 \bmod 12. \end{cases}$$

2.2.2 Isogenies

Next, we introduce the protagonist of this thesis: *isogenies* between elliptic curves.

Definition 2.2.11. Let $\varphi : E \to E'$ be a map between two elliptic curves E, E' defined over k. The map φ is called *isogeny* if it is a non-constant morphism of projective varieties mapping $\mathcal{O}_E$ to $\mathcal{O}_{E'}$ defined over an extension of k. Two curves E, E' are said to be *isogenous* if there exists an isogeny between them. We denote the set of all isogenies over k from E to E' together with the zero-map by $\mathrm{Hom}_k(E, E')$.

Note that any set $\mathrm{Hom}_k(E, E')$ inherits the structure of an abelian group through the *sum of two isogenies* given by

$$(\varphi + \psi)(P) = \varphi(P) + \psi(P). \tag{2.4}$$

For isogenies defined over $\overline{k}$, we will sometimes skip the subscript and we write $\mathrm{Hom}(E, E')$ for $\mathrm{Hom}_{\overline{k}}(E, E')$.

We may equivalently define isogenies as non-constant rational maps between the elliptic curves E and E' that are also group homomorphisms [Sil09, Sect. III.4]. We say that an isogeny is *defined over* k if its expression as a rational map can be defined with coefficients in k.

Let E be an elliptic curve defined by the equation $f(X, Y) = 0$ over k. The ring of regular functions on E is defined as $k[E] := k[X, Y]/\langle f \rangle$. Its field of fractions is called the *function field* of E, which we denote by $k(E)$.

Definition 2.2.12. An isogeny $\varphi : E \to E'$ induces an embedding of the function field $k(E')$ in $k(E)$ by composition,

$$\varphi^* : k(E') \to k(E),\ f \mapsto f \circ \varphi,$$

which we call the *pullback*.

Definition 2.2.13. Let $\varphi : E \to E'$ be an isogeny over k, and let $k(E)$, $k(E')$ be the function fields of E, E'. The *degree* of φ, denoted by $\deg \varphi$, is the degree of the extension $k(E)/\varphi^*(k(E'))$. The isogeny φ is called *separable* (respectively *inseparable*) if the extension of function fields is separable (respectively inseparable).

If $\varphi : E \to E'$ is an isogeny of degree D, we will often say that φ is a *D-isogeny* and that E and E' are *D-isogenous*.

Note that degrees of field extensions are multiplicative and therefore we have

$$\deg(\varphi \circ \psi) = \deg(\varphi) \cdot \deg(\psi).$$

Frobenius. Let E be an elliptic curve over a finite field of characteristic p. The *p^r-power Frobenius* is the map

$$\pi^r : (x, y) \mapsto (x^{p^r}, y^{p^r}), \tag{2.5}$$

which is a morphism of degree p^r from E to E^{p^r}. One can easily check that the equation of the curve E^{p^r} is obtained by raising all coefficients of the equation of E to the power of p^r. In particular, we have $E^{p^r} = E$ whenever E is defined over $\mathbb{F}_{p^r}$.

In cryptography, we will always work over finite fields, say of characteristic p. In those cases, any isogeny of degree coprime to p is separable of degree equal to the cardinality of its kernel, and the only inseparable isogenies are of the form π^r for some $r \in \mathbb{Z}$ (up

to composition with an isomorphism) of degree equal to p^r. Further, every isogeny can be written as a composition of a separable isogeny and π^r for some $r \in \mathbb{Z}$ (see [Sil09, II.2.12]).

By the following proposition, every finite subgroup of the domain curve gives rise to an isogeny with this subgroup as its kernel.

Proposition 2.2.14 ([Sil09, III.4.12])**.** *Let E/k be an elliptic curve and $G \subset E(\overline{k})$ be a finite subgroup that is $\mathsf{Gal}(\overline{k}/k)$-invariant. Up to k-isomorphism, there is a unique elliptic curve E'/k and a separable isogeny $\varphi : E \to E'$ over k satisfying* $\ker(\varphi) = G$.

This justifies the notation E/G for the codomain of an isogeny from E with kernel G. Since every isogeny is a group homomorphism, we have a bijection between separable isogenies up to $\overline{k}$-isomorphism and finite subgroups of E defined over $\overline{k}$. Because of this correspondence, we sometimes call an isogeny *cyclic* if its kernel is a cyclic subgroup.

Moreover, a chain of proper subgroups corresponds to a chain of isogenies.

Corollary 2.2.15. *Let E be an elliptic curve and $G \subset E(\overline{k})$ be a finite subgroup. For every subgroup of $G' \subset G$, the isogeny $\varphi : E \to E/G$ can be decomposed as*

$$\varphi : E \xrightarrow{\phi_1} E/G' \xrightarrow{\phi_2} E/G,$$

where $\ker(\phi_1) = G'$ *and* $\ker(\phi_2) = \phi_1(G)$.

Pushforward and pullback. Let A, B be coprime integers. Any isogeny $E \to E'$ of degree AB can be decomposed in two ways as $\varphi'_A \circ \varphi_B$ or $\varphi'_B \circ \varphi_A$, where φ_A, φ'_A (resp. φ_B, φ'_B) have degree A (resp. B). This creates a commutative diagram depicted in Fig. 2.2, where $\ker \varphi'_A = \varphi_B(\ker \varphi_A)$ and $\ker \varphi'_B = \varphi_A(\ker \varphi_B)$. Given φ_A and φ_B we define φ'_A (resp. φ'_B) as the *pushforward* of φ_A through φ_B (resp. φ_B through φ_A), which we denote by $\varphi'_A = [\varphi_B]_*\varphi_A$ (resp. $\varphi'_B = [\varphi_A]_*\varphi_B$). This is the dual notion of the *pullback*, $[\cdot]^*\cdot$, introduced in Definition 2.2.12, where $\varphi_A = [\varphi_B]^*\varphi'_A$ and $\varphi_B = [\varphi_A]^*\varphi'_B$.

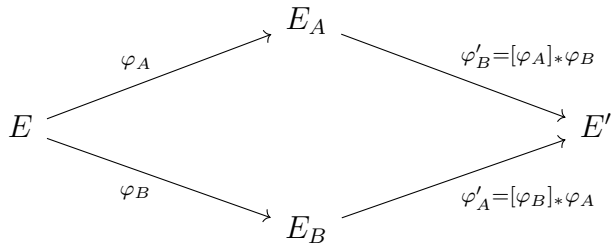

Fig. 2.2 Commutative diagram depicting the decomposition of an isogeny of degree AB.

Finally, it turns out that being isogenous is an equivalence relation.

Proposition 2.2.16. *Let $\varphi : E \to E'$ be an isogeny. Then, there is a unique isogeny $\widehat{\varphi} : E' \to E$ such that $\widehat{\varphi} \circ \varphi = [\deg \varphi]_E$ called the* dual *of φ. The dual has the following properties:*

- $\deg \varphi = \deg \widehat{\varphi}$
- $\widehat{\varphi + \psi} = \widehat{\varphi} + \widehat{\psi}$ *for any isogeny* $\psi : E \to E'$
- $\widehat{\psi \circ \varphi} = \widehat{\varphi} \circ \widehat{\psi}$ *for any isogeny* $\psi : E' \to E''$
- $\widehat{\widehat{\varphi}} = \varphi$ *and* $\widehat{[m]} = [m]$ *for all* $m \in \mathbb{Z}$.

Proof. See Theorems III.6.1 and III.6.2 in [Sil09]. □

From this it is easy to see that $\ker(\widehat{\varphi})$ is the subgroup $\varphi(E[\deg \varphi])$ of E' using the notation of the proposition.

While the problem of computing isogenies between two given curves over finite fields is considered hard and at the heart of isogeny-based cryptography, Schoof's point counting algorithm [Sch85] and the following theorem due to Tate allow us to efficiently decide whether two elliptic curves are isogenous.

Theorem 2.2.17 (Tate [Tat66])**.** *Two elliptic curves $E/\mathbb{F}_q$, $E'/\mathbb{F}_q$ are isogenous over $\mathbb{F}_q$ if and only if $\#E(\mathbb{F}_q) = \#E'(\mathbb{F}_q)$.*

By Proposition 2.2.9, one implication of this theorem is that elliptic curves connected by isogenies will either all be ordinary or all be supersingular.

Computing isogenies. After Proposition 2.2.14 established the correspondence between kernels and isogenies, a natural question is whether one can explicitly compute an isogeny given a starting curve and a kernel. This was answered by Vélu [Vél71].

Proposition 2.2.18 (Vélu's formulae)**.** *Let $E : y^2 = x^3 + ax + b$ be an elliptic curve over a field k in Weierstraß form and let $H \subset E(\overline{k})$ be a finite subgroup. For any function $\tau \in k(E)$ and point $P \in E$, define*

$$f_\tau(P) := \tau(P) + \sum_{\substack{Q \in H \\ Q \neq \mathcal{O}_E}} \left(\tau(P+Q) - \tau(Q)\right).$$

Let $x, y \in k(E)$ be the projections to the Weierstraß coordinates on E. Then the map

$$\varphi : E \to E/H, P \mapsto \left(f_x(P), f_y(P)\right),$$

where poles of f_x, f_y are mapped to the point at infinity, is a separable isogeny with kernel H to an elliptic curve in Weierstraß form.

Remark 2.2.19. Given three points on E/H computed using Vélu's formulae, we can easily retrieve the curve's Weierstraß model.

For the remainder of the thesis, we will only be interested in computing and representing isogenies between elliptic curves defined over finite fields.

An explicit computational example can be found in the section about Vélu's formulae in Galbraith's book [Gal12]. Since the degree of a separable isogeny is equal to the cardinality of its kernel, we see that naïvely evaluating Vélu's formulae for an isogeny φ of degree D has complexity $O(D)$. By now there are different algorithms to compute a separable isogeny, but they are all based on Vélu's formulae. Recently, Bernstein, De Feo, Leroux and Smith designed a variant, called $\sqrt{\text{élu}}$, which can evaluate an isogeny φ of degree D asymptotically with complexity $O^*(\sqrt{D})$ operations over the field of definition of $\ker\varphi$. Due to the constants, this new variant outperforms classical Vélu's formulae only for $D \geq 113$ [BDLS20, Appx. A.3].

By Corollary 2.2.15, we can decompose any isogeny into prime-degree isogenies and apply Vélu's formulae to each step separately. Thus the efficiency of any isogeny computation mainly depends on the largest prime factor ℓ dividing the degree of the isogeny and the size of the field extension containing $E[\ell] \cap \ker\varphi$.

Representing isogenies. In practice, we represent cyclic D-isogenies φ with domain E, i.e. isogenies where the kernel is a cyclic subgroup of cardinality D of E, as tuples (E, P), where P generates $\ker\varphi$. We call this a *kernel representation*.

In general, for an elliptic curve E/k, a subgroup of $E(\overline{k})$ of size ℓ will only be defined over a field extension of exponential degree in $\log(\ell)$ over k. To write down the kernel of large degree isogenies efficiently, one variant is to choose isogenies that are of powersmooth degree and thus their kernel can be written as a direct product of subgroups of coprime cardinality each represented by a generator defined over a small enough extension field. The isogeny can then be computed using Corollary 2.2.15, pushing each subgroup used as the kernel of the following step through all the previously computed isogenies. This ensures that we will only need to work over a field big enough to contain the direct product of any pairs of subgroups instead of the entire kernel at once. Another variant more commonly used in isogeny-based cryptography is to work with elliptic curves with a large enough smooth subgroup over a small extension field. For example, taking a supersingular curve defined over $\mathbb{F}_{p^2}$, where the prime p is of the form $p = s - 1$ for some smooth s, ensures the s-torsion is defined over $\mathbb{F}_{p^2}$ by Proposition 2.2.9.

Using techniques similar to compression in the SIDH key exchange [AJK+16, CJL+17, ZSP+18, NR19], which we will introduce in Section 2.3.2, this representation can be compressed to $O(\log(p) + \log(D))$ bits, even when the generator P is defined over a large field extension of $\mathbb{F}_{p^2}$. This compression is relevant when large degree isogenies are exchanged as part of a cryptographic protocol such as a key exchange or a digital signature.

2.2.3 Endomorphism rings

Next, we introduce a mathematical object that plays an important role in isogeny-based cryptography as it carries a lot of information about an elliptic curve.

Definition 2.2.20. An isogeny from a curve to itself is called an *endomorphism*. Let $\mathrm{End}(E) := \mathrm{Hom}(E, E) \cup \{[0]\}$ be the set of all isogenies from E to itself together with the multiplication-by-zero map. With addition inherited from pointwise addition of endomorphisms (see Eq. (2.4)) and multiplication being composition, this set has the structure of a ring, called the *endomorphism ring*.

Since scalar multiplication exists on every elliptic curve, we can embed $\mathbb{Z}$ as a subring of $\mathrm{End}(E)$ for any elliptic curve E. We will often identify $\mathbb{Z}$ with scalar multiplication and just write m instead of $[m]$ in the following. Mapping the non-zero endomorphisms to their dual and the zero-map to itself defines an involution $\bar{}\,: \mathrm{End}(E) \to \mathrm{End}(E)$, usually referred to as *conjugation*.

Identifying scalar multiplication $[m]$ with $m \in \mathbb{Z}$, we know from Proposition 2.2.16 that for all $\theta \in \mathrm{End}(E)$, we have $\theta\overline{\theta} = \deg(\theta) \in \mathbb{Z}$. Further, using

$$\begin{aligned} \deg(\theta) &= \theta\widehat{\theta} \\ \deg(\theta + [1]) &= (\theta + [1])\widehat{(\theta + [1])} = \theta\widehat{\theta} + \theta + \widehat{\theta} + [1], \end{aligned}$$

where $[1]$ denotes the identity map and using that conjugates correspond to duals, one can verify that

$$\deg(\theta + 1) - \deg(\theta) - 1 = (\theta + \overline{\theta}) \in \mathbb{Z}.$$

Since every $\theta \in \mathrm{End}(E)$ satisfies the quadratic equation $\theta^2 = (\theta + \overline{\theta})\theta - \overline{\theta}\theta$, which is over the integers by the above, θ can be viewed as an algebraic integer.

Definition 2.2.21. Let $\theta \in \mathrm{End}(E)$. Then $N(\theta) := \theta\overline{\theta}$ and $\mathrm{tr}(\theta) := \theta + \overline{\theta}$ are called the *norm* and *trace* of θ, respectively.

We have already mentioned that we can embed $\mathbb{Z}$ as a subring of $\mathrm{End}(E)$ for any elliptic curve E via scalar multiplication. If E is defined over a field of characteristic zero, then this may already be the entire endomorphism ring. On the other hand, we *always* have non-scalar endomorphisms if we look at elliptic curves defined over finite fields.

Let E be an elliptic curve defined over a finite field $\mathbb{F}_{p^r}$ of characteristic p. Then, we know the p^r-power Frobenius (2.5) is an isogeny from E to E, i.e. an endomorphism. Let $\#E(\mathbb{F}_{p^r}) = p^r + 1 - t$, then the characteristic polynomial of π^r is $(\pi^r)^2 - t\pi^r + p^r$. If π^r acts like a non-scalar endomorphism, which is typically the case, then $\mathbb{Z}[\pi^r] \cong \mathbb{Z}[\sqrt{t^2 - 4p^r}]$.

For some curves over finite fields, this is already the entire endomorphism ring. However, the endomorphism ring could be larger, e.g. if some d divides $\pi^r - c \in \mathbb{Z}[\pi^r]$ for $c, d \in \mathbb{Z}$, then $\mathbb{Z}[\pi^r] \subsetneq \mathbb{Z}[\frac{\pi^r - c}{d}]$ is contained in the endomorphism ring, or if the endomorphism ring has rank larger than 2. In the cases where we deal with endomorphism rings of larger rank, the following algebraic object will play an important role.

Definition 2.2.22. Let p be a prime number and let

$$(a, b) = \begin{cases} (-1, -1), & \text{if } p = 2 \\ (-1, -p), & \text{if } p \equiv 3 \bmod 4 \\ (-q, -p), & \text{if } p \equiv 1 \bmod 4 \end{cases}$$

where $q \equiv 3 \bmod 4$ is a prime that is not a square modulo p, guaranteed to exist by Dirichlet's theorem on primes in arithmetic progressions. The four-dimensional $\mathbb{Q}$-algebra spanned by $1, i, j, ij$ with multiplication rules

$$i^2 = a, \quad j^2 = b, \quad \text{and } ij = -ji,$$

is called the *quaternion algebra ramified at p and ∞*, denoted $B_{p,\infty}$. Up to isomorphism $B_{p,\infty}$ is independent of the choice made for (a, b).

For more background on quaternion algebras, we refer to Voight's book [Voi21]. Further, throughout the thesis we will use the following definition for an order (in quadratic fields or in quaternion algebras).

Definition 2.2.23. Let A be a finite-dimensional algebra over $\mathbb{Q}$. An *order* in A is a subring of A that is a free abelian group generated over $\mathbb{Z}$ by a $\mathbb{Q}$-basis for A. The orders in A are partially ordered by inclusion. Let $\mathcal{O}$, $\mathcal{O}'$ be orders in A, then $\mathcal{O}'$ is called a *superorder* of $\mathcal{O}$, if $\mathcal{O} \subsetneq \mathcal{O}'$. An order in A is called *maximal* if it has no superorder.

Now, we can state a theorem that classifies all possible structures that can appear for an endomorphism ring of an elliptic curve.

Theorem 2.2.24. *Let E/k be an elliptic curve.*

- *If $char(k) = 0$, then $\mathrm{End}(E) \cong \mathbb{Z}$ or $\mathrm{End}(E)$ is isomorphic to an order in an imaginary quadratic field.*
- *If $char(k) = p > 0$ and E is an ordinary curve, then $\mathrm{End}(E)$ is isomorphic to an order in an imaginary quadratic field.*
- *If $char(k) = p > 0$ and E is a supersingular curve, then $\mathrm{End}(E)$ is isomorphic to a maximal order in a quaternion algebra ramified at p and at infinity.*

Proof. See e.g. [Sil09, III. Cor. 9.4 and Rem. 9.4.1] for a proof that $\mathrm{End}(E)$ is $\mathbb{Z}$, an order in an imaginary quadratic field or an order in the quaternion algebra ramified at p and at infinity. The distinction in the case of positive characteristic is due to Deuring [Deu41]. □

Example 2.2.25. Let $p \equiv 3 \bmod 4$ be a prime and let E be the elliptic curve over $\mathbb{F}_p$ given by the affine equation $y^2 = x^3 + x$. Clearly, the Frobenius $\pi : E \to E, (x, y) \mapsto (x^p, y^p)$ lies in the endomorphism ring. Note that we have $\sqrt{-1} \in \mathbb{F}_{p^2} \setminus \mathbb{F}_p$ for $p \equiv 3 \bmod 4$. Looking at morphisms defined over $\mathbb{F}_{p^2}$, we can then check that $\iota : E \to E, (x, y) \mapsto (-x, \sqrt{-1} \cdot y)$ is an automorphism of order four.

We have $\iota^2 = -1$, $\pi^2 = -p$, and $\iota\pi = -\pi\iota$ as $\sqrt{-1}^p = -\sqrt{-1}$. The $\mathbb{Z}$-module generated by $1, \iota, \pi, \iota\pi$ is contained in $\mathrm{End}(E)$, and thus E must be supersingular. While this $\mathbb{Z}$-module is not isomorphic to a maximal order in the quaternion algebra, the full endomorphism ring is only slightly larger and generated by $1, \iota, \frac{\iota+\pi}{2}$, and $\frac{1+\iota\pi}{2}$ (see e.g. [Sil09]).

The endomorphism ring carries not only a lot of information about the elliptic curve, but it also reveals information about its location in the isogeny graph, which we will introduce shortly. Further, the correspondence between ideals of the endomorphism ring and isogenies that we will describe in Section 2.2.5 is crucial for some constructions in isogeny-based cryptography.

2.2.4 Orientations of elliptic curves

Next, we introduce the concept of orientations of elliptic curves, which was introduced by Colò and Kohel [CK19]. Orientations provide a convenient framework to unify different group actions on ordinary and supersingular elliptic curves that were previously treated separately. For the definition, it is convenient to consider a slightly coarser object than

$\mathrm{End}(E)$, namely $\mathrm{End}(E) \otimes_{\mathbb{Z}} \mathbb{Q}$, which is usually referred to as the *endomorphism algebra*. It consists of elements of the form θ/d, where $\theta \in \mathrm{End}(E)$ and $d \in \mathbb{Z} \setminus \{0\}$.

Definition 2.2.26. Let k be an algebraically closed field and $\mathcal{O}$ an order in an imaginary quadratic field K. A *K-orientation* on an elliptic curve E/k is an injective ring homomorphism $\iota : K \hookrightarrow \mathrm{End}(E) \otimes_{\mathbb{Z}} \mathbb{Q}$. For an order $\mathcal{O} \subset K$, we say that a K-orientation is an *$\mathcal{O}$-orientation*, if $\iota(\mathcal{O}) \subset \mathrm{End}(E)$, i.e. $\iota : \mathcal{O} \hookrightarrow \mathrm{End}(E)$. If ι cannot be extended to a strict superorder $\mathcal{O}' \supsetneq \mathcal{O}$ in K then the $\mathcal{O}$-orientation is called *primitive*.

If ι is a K-orientation on E (resp. primitive $\mathcal{O}$-orientation), a pair (E, ι) is called a *K-oriented* (resp. *primitive $\mathcal{O}$-oriented*) *elliptic curve*.

It is easy to see that every K-orientation ι induces a primitive $\mathcal{O}$-orientation for a unique order $\mathcal{O} \subset K$, namely for $\mathcal{O} = \iota^{-1}(\mathrm{End}(E))$. Further, every non-scalar endomorphism $\theta \in \mathrm{End}(E)$ naturally gives rise to an orientation. Indeed, we know from Section 2.2.3 that $\theta^2 - \mathrm{tr}(\theta) + N(\theta) = 0$. Fixing

$$\sigma := \frac{\mathrm{tr}(\theta) + \sqrt{\mathrm{tr}(\theta)^2 - 4N(\theta)}}{2} \in \mathbb{C} \tag{2.6}$$

we obtain an orientation $\iota : \mathbb{Z}[\sigma] \hookrightarrow \mathrm{End}(E)$, which is unique if we impose that $\iota(\sigma) = \theta$. Conversely, every orientation arises in this way. The orientation ι extends uniquely to a primitive orientation $\iota' : \mathcal{O}' \hookrightarrow \mathrm{End}(E)$, under which the superorder $\mathcal{O}' \subseteq \mathbb{Q}(\sigma)$ of $\mathcal{O}$ is mapped isomorphically to the subring

$$\mathrm{End}(E) \cap \mathbb{Q}(\theta) := \{\, \omega \in \mathrm{End}(E) \mid \exists\, r \in \mathbb{Z} \setminus \{0\} : r\omega \in \mathbb{Z}[\theta] \,\} \subset \mathrm{End}(E).$$

Example 2.2.27. Let E be an ordinary elliptic curve over a finite field $\mathbb{F}_{p^r}$. Then we can apply (2.6) to the Frobenius endomorphism π^r, which yields an orientation

$$\iota : \mathbb{Z}[\sigma] \to \mathrm{End}(E), \sigma \mapsto \pi^r, \qquad \text{with } \sigma = \frac{t + \sqrt{t^2 - 4p^r}}{2}$$

where t is the trace of Frobenius. Since $\mathrm{End}(E) \cap \mathbb{Q}(\pi^r) = \mathrm{End}(E)$, the induced primitive orientation is an isomorphism.

Example 2.2.28. If E is a supersingular elliptic curve over a finite prime field $\mathbb{F}_p$ with $p > 3$, then we have $\pi^2 = -p$. So in this case we obtain an orientation

$$\iota : \mathbb{Z}[\sqrt{-p}] \to \mathrm{End}(E), \sqrt{-p} \mapsto \pi.$$

The image $\mathrm{End}(E) \cap \mathbb{Q}(\pi)$ of this primitive orientation equals $\mathrm{End}_p(E)$, the ring of $\mathbb{F}_p$-rational endomorphisms of E.

Supersingular elliptic curves have endomorphism rings isomorphic to an order in a quaternion algebra by Theorem 2.2.24. Thus, they admit infinitely many primitive orientations. At the other extreme, an elliptic curve defined over a field of characteristic zero does not admit any orientations at all, unless it has complex multiplication, i.e. its endomorphism ring is larger than $\mathbb{Z}$.

Definition 2.2.29. Let (E, ι) be an $\mathcal{O}$-oriented elliptic curve and let $\varphi : E \to F$ be an isogeny. We can define an $\mathcal{O}$-orientation $\varphi_*(\iota)$ on F by

$$\varphi_*(\iota)(\alpha) := \frac{1}{\deg(\varphi)} \varphi \circ \iota(\alpha) \circ \widehat{\varphi}, \quad \forall \alpha \in \mathcal{O}.$$

Given two $\mathcal{O}$-oriented elliptic curves (E, ι_E) and (F, ι_F), we call an isogeny $\varphi : E \to F$ *$\mathcal{O}$-oriented*, if $\iota_F = \varphi_*(\iota_E)$. In this case, we write $\varphi : (E, \iota_E) \to (F, \iota_F)$. We call two $\mathcal{O}$-oriented curves (E, ι_E) and (F, ι_F) *isomorphic*, if there exists an isomorphism $\varphi : E \to F$ such that $\varphi_*(\iota_E) = \iota_F$.

2.2.5 Class group actions on oriented elliptic curves

Let $\mathcal{O}$ be an order in an imaginary quadratic field. The set of invertible fractional $\mathcal{O}$-ideals quotiented by the subgroup of principal fractional ideals is an abelian group under ideal multiplication that is called the *class group of* $\mathcal{O}$, denote by $\mathrm{Cl}(\mathcal{O})$. Let

$$\mathrm{Ell}_{\overline{k}}^{\mathrm{all}}(\mathcal{O}) := \{ (E, \iota) \mid E \text{ ell. curve over } \overline{k}, \iota \text{ primitive } \mathcal{O}\text{-orientation on } E \} / \cong$$

be the set of all primitively $\mathcal{O}$-oriented elliptic curves over a field $\overline{k}$ up to isomorphism as defined in Definition 2.2.29. This set comes equipped with a group action by the ideal class group of $\mathcal{O}$. For $k = \mathbb{C}$, this is a classical result from the theory of complex multiplication. The case where k is the algebraic closure of a finite field is treated in detail in [Wat69]. Next, we describe this action.

Let (E, ι) be a primitively $\mathcal{O}$-oriented elliptic curve and $\mathfrak{a} \subset \mathcal{O}$ an invertible ideal of norm coprime to $\max\{1, \mathrm{char}(k)\}$. One defines the *$\mathfrak{a}$-torsion subgroup* as

$$E[\mathfrak{a}] = \bigcap_{\alpha \in \mathfrak{a}} \ker \iota(\alpha). \tag{2.7}$$

There exists an elliptic curve F and a separable isogeny $\varphi : E \to F$ with $\ker(\varphi) = E[\mathfrak{a}]$, which is unique up to k-isomorphism. We denote the k-isomorphism class of $(F, \varphi_*(\iota))$

by $[\mathfrak{a}](E,\iota)$. For a proof that $E[\mathfrak{a}]$ is indeed a finite subgroup of E, see [Wat69]. We may sometimes write $\varphi_{\iota(\mathfrak{a})}$ for the isogeny $E \to F$, or just $\varphi_{\mathfrak{a}}$ if the orientation is implicit.

To compute $E[\mathfrak{a}]$, note that it suffices to compute the intersection given in Eq. (2.7) on a set of generators of $\mathfrak{a}$.

Consider (2.7) again for a primitively $\mathcal{O}$-oriented elliptic curve (E,ι) and let $\theta \in \mathcal{O}$. Then, $E[\mathfrak{a}\theta] = \iota(\theta)^{-1}E[\mathfrak{a}]$ contains $\ker(\iota(\theta))$ by construction. Thus, $\varphi_{\mathfrak{a}\theta}$ decomposes as the *endomorphism* $\ker(\iota(\theta))$ and $\varphi_{\mathfrak{a}}$. Hence, the isogenies with kernels $E[\mathfrak{a}]$ and $E[\mathfrak{a}\theta]$ have codomains that are isomorphic. Similarly one sees that both isogenies induce the same orientation up to isomorphism.

Proposition 2.2.30. *Let (E,ι) be an oriented elliptic curve over k with primitive $\mathcal{O}$-orientation, $\mathfrak{a} \subset \mathcal{O}$ an integral ideal and $\theta \in \mathcal{O}$. Then*

$$[\mathfrak{a}](E,\iota) \cong_k [\mathfrak{a}\theta](E,\iota).$$

In particular, the k-isomorphism class of $[\mathfrak{a}](E,\iota)$ depends only on the ideal class of $\mathfrak{a} \in \mathrm{Cl}(\mathcal{O})$, where $\mathrm{Cl}(\mathcal{O})$ denotes the ideal class group of $\mathcal{O}$.

This gives rise to a group action by the ideal class group $\mathrm{Cl}(\mathcal{O})$ on the set of primitively $\mathcal{O}$-oriented elliptic curves. This group action is free, i.e. no two ideal classes act the same on any primitive oriented elliptic curve (see e.g. [Wat69] for the case of ordinary elliptic curves over finite fields naturally oriented by Frobenius or [Onu21] for the general oriented case). However, in general it will not be transitive on the set of all primitively $\mathcal{O}$-oriented elliptic curves, see e.g. [Sch87, Thm. 4.5] and [Onu21, Prop. 3.3] for some specific examples with two orbits. However, we have the following result [CK19, Onu21].

Theorem 2.2.31. *Let k be a field of characteristic $p \geq 0$, and let $\mathcal{O}$ be an order in an imaginary quadratic field K such that $Ell_{\overline{k}}^{all}(\mathcal{O})$ is non-empty. Then*

$$\begin{aligned} \mathrm{Cl}(\mathcal{O}) \times Ell_{\overline{k}}^{all}(\mathcal{O}) &\to Ell_{\overline{k}}^{all}(\mathcal{O}) \\ ([\mathfrak{a}],(E,\iota)) &\mapsto [\mathfrak{a}](E,\iota), \end{aligned}$$

where $\mathfrak{a} \subset \mathcal{O}$ is an integral representative of its ideal class, is a well-defined and free group action with at most two orbits.

In isogeny-based cryptography, we are typically in the case where we want to find an isogeny between two primitively $\mathcal{O}$-oriented curves knowing that such an isogeny exists. Hence, we can assume that the oriented elliptic curves we are dealing with are located in the same orbit of the class group action. Restricting the group action to one orbit, say $\mathrm{Ell}_{\overline{k}}(\mathcal{O}) \subset \mathrm{Ell}_{\overline{k}}^{\mathrm{all}}(\mathcal{O})$, simplifies our exposition and allows us to assume transitivity.

2.2.6 Deuring's correspondence

By Theorem 2.2.24, we know that the endomorphism ring of a supersingular elliptic curve is a maximal order in a quaternion algebra. One may wonder whether every maximal order in a quaternion algebra appears as an endomorphism ring of a supersingular elliptic curve, or how many supersingular elliptic curves correspond to the same maximal order. This question was answered by a seminal paper due to Deuring [Deu41], which also provides the proofs for all of the remaining statements in this section.

Theorem 2.2.32 (The Deuring correspondence)**.** *There is a one-to-one correspondence between j-invariants over $\mathbb{F}_{p^2}$ of supersingular elliptic curves up to Galois conjugacy and maximal orders $\mathcal{O}$ of $B_{p,\infty}$ up to isomorphism. Hereby, $\{j(E), j(E)^p\}$, where $E/\mathbb{F}_{p^2}$ is a supersingular elliptic curve, corresponds to* $\mathrm{End}(E)$*.*

Note that when E is defined over $\mathbb{F}_{p^2}$, then the Frobenius $\pi : E \to E^p$ is an isogeny such that $j(E)^p = j(E^p)$. By the Deuring correspondence, there are at most two supersingular elliptic curves having isomorphic endomorphism rings. If this is the case, their j-invariants are conjugates in $\mathbb{F}_{p^2}/\mathbb{F}_p$ and both curves are connected by the Frobenius isogeny $\pi : E \to E^p$.

The correspondence is even more explicit, as it allows us to translate isogenies to ideals and vice versa. We get something similar to the class group action on the set of elliptic curves with a certain primtive orientation described in the previous section. To make this more precise, we define the following.

Definition 2.2.33. We call two fractional left ideals of an order $\mathcal{O} \subset B_{p,\infty}$ *equivalent*, if there exists a unit $\alpha \in B_{p,\infty}$ such that $J = I\alpha$. We denote the set of all fractional left ideals in $\mathcal{O}$ equivalent to I by $[I]$ and we call it the *class of* I.

For any ideal, we can further define a non-commutative analogue of (2.7).

Proposition 2.2.34. *Let E be a supersingular elliptic curve and* $\mathrm{End}(E) \cong \mathcal{O} \subset B_{p,\infty}$*. Any non-zero left ideal I of $\mathcal{O}$ defines the following finite subgroup of E:*

$$E[I] := \bigcap_{\alpha \in I} \ker(\alpha).$$

We will sometimes denote the isogeny $E \to E/E[I]$ by φ_I. If I is a principal ideal, then $\varphi_I : E \to E/E[I]$ is an endomorphism. Further, when taking two equivalent ideals I, J in the same ideal class, we have $E/E[I] \cong E/E[J]$, since $E \to E[J]$ can be factored as the isogeny $E \to E[I]$ precomposed with an endomorphism. Theorem 2.2.31 has the following non-commutative analogue.

Proposition 2.2.35. *Let E_0 be a supersingular elliptic curve over $\mathbb{F}_{\overline{p}}$ and let* $\mathrm{End}(E_0)$ *be isomorphic to $\mathcal{O}_0 \subset B_{p,\infty}$. For every supersingular elliptic curve $E/\mathbb{F}_{\overline{p}}$, there exists a unique left ideal class $[I] \subset \mathcal{O}_0$ such that E is isomorphic to the codomain of the isogeny with kernel $E_0[J]$, as defined in Proposition 2.2.34, for any representative $J \in [I]$. The endomorphism ring of the codomain E is isomorphic to the right order of J in $B_{p,\infty}$.*

Remark 2.2.36. Proposition 2.2.35 also provides one way to see that all supersingular elliptic curves over the same field are isogenous.

Note that while the codomains of isogenies corresponding to equivalent ideals are isomorphic, the isogenies will in general not be the same. In fact, one can show that the degree of the isogeny equals the norm of the corresponding ideal. Further, the dual isogeny corresponds to the conjugate of an ideal and composition of isogenies corresponds to the product of ideals [Deu41].

2.2.7 Isogeny graphs

We know that we can easily determine whether two elliptic curves are isogenous by Theorem 2.2.17. However, for many applications one wants to know *how* the curves are connected. To gather more information about this, it is natural to study the structure of the following objects.

Definition 2.2.37 (ℓ-isogeny graphs)**.** Let k be a field and ℓ be a prime not divisible by the characteristic of k. The *ℓ-isogeny graph* is the graph where vertices are k-isomorphism classes of elliptic curves over k. The edges represent isogenies over k of degree ℓ between curves in the corresponding isomorphism classes. Let K be an imaginary quadratic field. The *K-oriented* ℓ-isogeny graph is defined analogously with vertices being k-isomorphism classes of K-oriented elliptic curves and edges being K-oriented isogenies of degree ℓ.

Note that the existence of dual isogenies justifies the viewing of isogeny graphs as undirected graphs. Concerning the multiplicity of edges, one has to be a little careful when considering graphs containing the isomorphism class of curves with j-invariant 0 or 1728 due to additional non-trivial automorphisms of these curves, see e.g. [Sil09, Appx. A, Prop. 1.2].

Typically, we will restrict ourselves to a subgraph containing curves of a given order, which is a disconnected component of the full ℓ-isogeny graph by Theorem 2.2.17. Away from the two special vertices mentioned above, the connected components of the ℓ-isogeny graph turn out to have two possible shapes: the structured shape of a *volcano* or a less structured expander graph which we refer to as the *full supersingular isogeny graph*. We

describe these graphs and in which context they occur in more detail next. For a nice illustration how the volcanoes containing the class of curves with j-invariant 0 or 1728 look like, we refer to Panny's thesis [Pan21, Fig. 2.3].

Volcanoes of oriented elliptic curves

Most of the results on isogeny volcanoes were developed for the special case of ordinary elliptic curves defined over a finite field using the Frobenius endomorphism to define an orientation as described in Example 2.2.27. Many of the results in this case originate from Kohel's thesis [Koh96] and for an overview of volcanoes of ordinary elliptic curves, we refer to [Sut13]. However, a lot of the theory generalises to the setting of oriented elliptic curves in general [CK19, Onu21].

Let K be an imaginary quadratic number field with maximal order $\mathcal{O}_K$ and let $\varphi : (E_1, \iota_1) \to (E_2, \iota_2)$ be a K-oriented ℓ-isogeny of K-oriented elliptic curves such that the domain and codomain are primitively oriented by $\mathcal{O}$ and $\mathcal{O}'$, respectively. Then one of the following is true [CK19]:

- $\mathcal{O} \subset \mathcal{O}'$ and $[\mathcal{O}' : \mathcal{O}] = \ell$, in this case φ is called *ascending*.
- $\mathcal{O} = \mathcal{O}'$, in this case φ is called *horizontal*.
- $\mathcal{O}' \subset \mathcal{O}$ and $[\mathcal{O} : \mathcal{O}'] = \ell$, in this case φ is called *descending*.

We use the following definition for volcanoes.

Definition 2.2.38. An *ℓ-volcano* is a connected undirected graph with vertices partitioned into levels $V_0, V_1 \ldots$, in which the subgraph on V_0, called the *crater*, is a regular graph of degree at most 2 and

a) For $i > 0$, each vertex in V_i has exactly one edge leading to a vertex in V_{i-1}, and every edge not on the crater is of this form.

b) Each vertex has degree $\ell + 1$, except for the vertices on the last level V_d in case the number of levels is finite (in which case V_d is called the *floor* and d is called the *depth* of the volcano).

The *K-oriented* ℓ-isogeny graph is the disjoint union of ℓ-volcanoes (see [CK19, Onu21]). For simplicity, assume we have only one ℓ-volcano. The crater consists of the finite set of primitive $\mathcal{O}_0$-oriented elliptic curves, where $\mathcal{O}_0 \subseteq K$ is *locally primitive* at ℓ, i.e. the index $[\mathcal{O}_K : \mathcal{O}_0]$ is coprime to ℓ. From elliptic curves with primitive $\mathcal{O}_0$-orientation, there cannot be an ascending isogeny. Descending in the volcano, the vertices on level V_i correspond to primitively $\mathcal{O}_i$-oriented elliptic curves such that $[\mathcal{O}_i : \mathcal{O}_{i+1}] = \ell$.

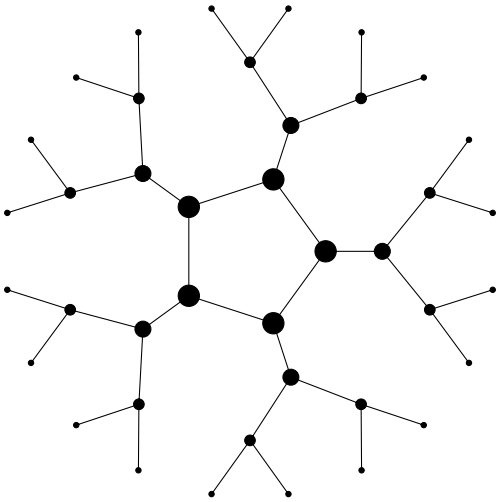

Fig. 2.3 2-volcano of $\mathcal{O}$-oriented curves (E, ι) for some $\mathcal{O} \subset K$ where $[\mathcal{O}_K : \mathcal{O}]$ has 2-adic valuation equal to 3. Curves on the crater have primitive $\mathcal{O}_0$-orientation, where $\mathcal{O}_0$ is locally primitive at $\ell = 2$, the ones on the floor are primitively $\mathcal{O}_3$-oriented and $[\mathcal{O}_0 : \mathcal{O}_3] = 2^3$.

Finite volcanoes. Instead of considering the full, infinite volcano of K-oriented curves, choosing a fixed order $\mathcal{O} \subset K$, we can consider the volcano consisting only of K-oriented curves (E, ι) with $\iota(\mathcal{O}) \subset \mathrm{End}(E)$. This restriction truncates the volcano of K-oriented curves at the level equal to the ℓ-adic valuation of the conductor of $\mathcal{O}$. That is, the resulting volcanoes are finite. Fig. 2.3 depicts such a finite volcano.

Isogeny volcanoes have found multiple applications such as the computation of endomorphism ring of ordinary elliptic curves [BS11] or the computation of Hilbert class polynomials [Sut11, BBEL08]. The "classical" isogeny volcanoes studied there arise as components of the ℓ-isogeny graph of curves defined over $\mathbb{F}_{p^r}$ with Frobenius orientation as in Example 2.2.27 truncated to those curves which contain $\mathbb{Z}[\pi^r]$ in their endomorphism ring.

For cryptography, however, a single volcano for a single fixed ℓ may not seem to be very useful on its own. Assuming that ℓ-isogenies can be computed efficiently, one can navigate the volcanoes and walk to the surface as described by Kohel [Koh96]. See [IJ13] for an alternative approach using the relation between the group structure of the rational ℓ-torsion on ordinary elliptic curves and the level in the volcano to navigate between different levels by the means of pairings. Being left with the crater, a single cycle or even disconnected, there is no distinction between walking a path between two vertices or retrieving it and thus the setting does not appear to be promising for public key cryptography.

Instead of considering just a single volcano for a fixed ℓ, a more common approach in cryptography is to consider a union of multiple volcanoes for different ℓ on the same set of vertices, as we will see in Section 2.3.1.

A different approach using oriented supersingular curves was taken in OSIDH [CK19], where the authors tried to reveal less information to make sure an adversary cannot recover an isogeny to the crater from a given oriented curve. Yet, Dartois and De Feo showed that the information provided was still too much for a wide range of parameters [DD22].

In Chapter 6, we present another idea where working in an ℓ-volcano with large prime ℓ prevents an adversary from efficiently computing the isogenies to the crater.

The full supersingular isogeny graph

Another graph important for cryptography, in particular for those cryptographic schemes that will play a major role in this thesis, is the supersingular ℓ-isogeny graph, where vertices are isomorphism classes of supersingular elliptic curves over $\overline{\mathbb{F}}_p$, which can be represented by a j-invariant in $\mathbb{F}_{p^2}$ (see Proposition 2.2.10). Given that the number of supersingular curves over $\overline{\mathbb{F}_p}$ up to isomorphism is roughly $p/12$, these graphs are interesting for cryptographic purposes only for large p, which we will assume to be the case in the following. Considering the torsion structure for the supersingular elliptic curves (Proposition 2.2.4) and that every subgroup of order ℓ of the elliptic curve defines an isogeny (Proposition 2.2.14) one can see that the ℓ-isogeny graph is $(\ell+1)$-regular (away from curves with j-invariants 0 or 1728 where the additional automorphism might change the edge's multiplicities).

Using the following proposition together with Proposition 2.2.35, one can further see that the supersingular ℓ-isogeny graphs form a connected component for every $\ell \neq p$.

Proposition 2.2.39. *[Voi21, Thm. 28.5.3] Let $\mathcal{O}$ be a maximal order in $B_{p,\infty}$ and let $I \subset \mathcal{O}$ be a non-zero left ideal. For any prime $\ell \neq p$, there exists an equivalent left ideal $J \in [I]$ whose norm is a power of ℓ.*

A heuristic polynomial time algorithm that makes Proposition 2.2.39 effective was provided by Kohel, Lauter, Petit and Tignol (KLPT) [KLPT14]. Later, Wesolowski gave a variant which relies only on the Generalised Riemann Hypothesis [Wes22b]. In Section 5.2.1, we give a brief sketch of the KLPT algorithm, but we refer to [KLPT14, DKL+20] for the technical details.

The fast mixing property of supersingular ℓ-isogeny graphs makes them interesting for cryptographic purposes. Indeed, Pizer showed that that ℓ-isogeny graphs are Ramanujan.

Theorem 2.2.40. *[Piz90, Thm. 1] Let p and ℓ be distinct primes with $\ell < p/4$. Then, the supersingular ℓ-isogeny graph over $\overline{\mathbb{F}_p}$ is a Ramanujan graph. Ramanujan means that all eigenvalues λ of the adjacency matrix satisfy $\lambda \leq 2\sqrt{\ell}$ which is asymptotically the best possible. In particular, the diameter of the graph is between $\log p$ and $2\log p$.*

Therefore, after taking only relatively few random steps in the ℓ-isogeny graph one can reach every vertex of the graph with roughly equal probability.

Despite containing all of the oriented ℓ-volcanoes of supersingular elliptic curves, the full supersingular ℓ-isogeny graph appears to be rather unstructured. Indeed, given a random supersingular elliptic curve it seems to be a hard problem to determine whether and where it lies on a specific volcano without first computing endomorphisms of this curve. At the time of writing this thesis, this is considered to be a computationally hard problem for supersingular elliptic curves defined over a finite field of sufficiently large characteristic p.

2.3 Isogeny-based key exchange protocols

In this section, we introduce the three main isogeny-based key exchange protocols and touch on some of their variants. First, we introduce the Couveignes–Rostovtsev–Stolbunov (CRS) and the CSIDH[1] key exchange protocols as instantiations of Diffie–Hellman key agreements from commutative group actions. The isogeny graphs that appear in this case are volcanoes. Note that we describe the schemes in the unifying framework of oriented elliptic curves which was not introduced until after CRS and CSIDH were proposed.

Second, we introduce the SIDH key exchange which uses isogeny walks in the full supersingular isogeny graph. This is the foundation of Chapters 3 to 6 which will all be concerned with various aspects of SIDH, its variants, or more advanced cryptographic protocols built on top of it.

2.3.1 CRS and CSIDH

Couveignes proposed the first isogeny-based key exchange but did not publish it at the time [Cou06]. The same idea was rediscovered independently by Rostovtsev and Stolbunov [RS06]. Couveignes and Rostovtsev–Stolbunov (CRS) suggested a post-quantum key exchange based on the group action of an ideal class group on a set of ordinary elliptic curves, which can be seen as a special case of the group action described

[1] pronounced “seaside”

in Section 2.2.5. We start by recalling how group actions can allow us to replace classical group-based Diffie–Hellman [DH76] by a (potentially) quantum-resistant alternative.

Diffie–Hellman key exchange from group actions

The analogue of the Diffie–Hellman key exchange for commutative group actions is as follows. Let $\cdot : G \times X \to X$ be a group action of an abelian group G on a set X. To agree on a shared secret over an insecure public channel, Alice and Bob first agree on a (public) element $E \in X$. Then, Alice and Bob each choose their secret $a \in G$ and $b \in G$, respectively, and exchange $a \cdot E$ and $b \cdot E$. Since a and b commute, both Alice and Bob can compute $(ab) \cdot E$ which acts as the shared secret.

Note that the classical Diffie–Hellman key exchange can also be seen as one arising from the action of $\mathbb{Z}/(p-1)\mathbb{Z}$ on the multiplicative *group* X of units $\mathbb{F}_p^*$. By removing the inherent structure of a group from X, it is hoped that for carefully chosen group actions the key exchange above remains secure even in the presence of quantum adversaries. For the key exchange to be secure, we require the following two problems to be hard.

Definition 2.3.1. Let $\cdot : G \times X \to X$ be a commutative, free and transitive group action. Given $E, E' \in X$, the *vectorisation problem* asks to recover $a \in G$ such that $a \cdot E = E'$.

The vectorisation problem seeks to invert the group action and thus can be seen as an analogue of the classical discrete logarithm problem. The analogue to the computational Diffie–Hellman problem is given by the following.

Definition 2.3.2. Let $\cdot : G \times X \to X$ be a commutative, free and transitive group action. Given $E, a \cdot E, b \cdot E$, the *parallelisation problem* asks to compute $(ab) \cdot E$.

For efficiently computable group actions, the vectorisation and the parallelisation problem were shown to be equivalent under a quantum polynomial time reduction [GPSV21].

A group action $\cdot : G \times X \to X$ is called a *hard homogenous space* if it can be computed efficiently and the vectorisation and parallelisation problems are hard [Cou06]. A more recent formalisation of properties to obtain cryptographically useful group actions was given by *Effective Group Actions* (EGA) in [ADMP20].

CRS

In the CRS key exchange, the group action based Diffie–Hellman is instantiated with the action of the ideal class group $\mathrm{Cl}(\mathcal{O})$ on the set of primitively $\mathcal{O}$-oriented ordinary elliptic curves defined over a finite field as described in Section 2.2.5 (restricting to one

orbit if necessary). While the general notion of oriented elliptic curves did not exist when CRS was proposed, the concrete orientation suggested was the Frobenius orientation on ordinary elliptic curves presented in Example 2.2.27. Note that $\mathrm{Cl}(\mathcal{O})$ is an abelian group which is necessary for the key exchange to work. Couveignes and Rostovtsev–Stolbunov conjectured the vectorisation and parallelisation problems to be computationally hard for this class group action.

Differences between Couveignes' and Rostovtsev–Stolbunov's proposals. For a fixed public curve E_0 with a primitive $\mathcal{O}$-orientation, Couveignes suggested that Alice and Bob sample their secrets $\mathfrak{a}$ and $\mathfrak{b}$ uniformly at random from $\mathrm{Cl}(\mathcal{O})$, respectively. To compute the curves $E_A := E_0/E_0[\mathfrak{a}]$ and $E_B := E_0/E_0[\mathfrak{b}]$, i.e. the group action of $\mathfrak{a}$ and $\mathfrak{b}$ on the starting curve, Couveignes suggested to translate the sampled ideals to an equivalent product of small-norm ideals. To do so, he proposed to compute the structure of $\mathrm{Cl}(\mathcal{O})$ using the Hafner–McCurley algorithm [HM89]. To avoid the expensive class group computation, Rostovtsev and Stolbunov proposed to instead sample the ideals $\mathfrak{a}$ and $\mathfrak{b}$ as products of small-norm prime ideals in the first place.

Next, we sketch the CRS key exchange following the approach suggested by Rostovtsev and Stolbunov.

Public parameters. Let p be a prime and $E_0/\mathbb{F}_p$ be an $\mathcal{O}$-oriented ordinary elliptic curve (with $\mathbb{Z}[\pi] \in \mathcal{O}$) and let $\ell_1, \ldots, \ell_n$ be a set of primes that split in $\mathcal{O}$. Let $\mathfrak{l}_i$ be fractional ideals above ℓ_i in $\mathcal{O}$ and let $\mathcal{I} \subset \mathbb{Z}$ a set of possible exponents.

Key generation. Alice samples a secret exponent vector $(e_i)_{1 \leq i \leq n} \in \mathcal{I}^n$, sets $\mathfrak{a} = [\prod_i \mathfrak{l}_i^{e_i}] \in \mathrm{Cl}(\mathcal{O})$ and computes her public key $E_A := E_0/E_0[\mathfrak{a}]$. Bob proceeds mutatis mutandis computing his public key $E_B := E_0/E_0[\mathfrak{b}]$.

Key exchange. Upon exchanging their public keys, Alice and Bob compute the curve $E_{AB} := E_0/E_0[\mathfrak{ab}]$ up to isomorphism as $E_B/E_B[\mathfrak{a}]$ and $E_A/E_A[\mathfrak{b}]$, respectively. The shared secret is the (hash of the) j-invariant of E_{AB}.

Fixing a set of small prime ideals above primes $\ell_1, \ldots, \ell_n$ that split in $\mathcal{O}$, one can think of the isogeny graph underlying the CRS key exchange as the union of 2-regular craters of ℓ_i-volcanoes all having the same set of vertices. A priori, it is not obvious that the union of such graphs is better connected than each of its components. However, it was shown that for sufficiently many ℓ_i and large finite fields the union is an expander graph [JMV09]. Thus, after taking only a few random steps along the ideals $\mathfrak{l}_i$ lying above the ℓ_i, one can reach any of the vertices in the graph with similar probability after only few steps.

Unfortunately, the CRS key exchange is too slow for practical use despite some work trying to accelerate it by De Feo, Kieffer and Smith [DKS18]. Their main goal was to ensure that $\#E_0(\mathbb{F}_p)$ is divisible by as many small ℓ_i as possible. For the ideals $\mathfrak{l}_i$ lying above such primes, the class group action could then be computed efficiently by evaluating Vélu's formulae over a small extension field. However, since finding ordinary elliptic curves with a specific number of points is considered a difficult problem, this approach only had limited success. However, the ideas helped develop CSIDH - a much faster version of a group action of a class group of an imaginary quadratic order on a set of *supersingular* elliptic curves.

CSIDH

As we pointed out before, the number of points on a supersingular elliptic curve over a finite field $\mathbb{F}_p$ is determined by the size of p. The core idea of CSIDH [CLM+18] is to use supersingular elliptic curves defined over a finite field $\mathbb{F}_p$ for a specific choice of p. Let $\mathcal{O} = \mathbb{Z}[\sqrt{-p}]$. Recall from Example 2.2.28, the p-power Frobenius, π, is an endomorphism for a supersingular elliptic curve defined over $\mathbb{F}_p$ and we obtain an $\mathcal{O}$-orientation on supersingular elliptic curves defined over $\mathbb{F}_p$ by identifying $\sqrt{-p}$ with π and $\mathbb{Z}$ with scalar-multiplication.

Let p be a prime of the form $4\ell_1 \cdots \ell_n \cdot f - 1$, where ℓ_i are small primes that split in $\mathcal{O}$ and $f \in \mathbb{Z}$ is a small cofactor, and let the starting curve E_0 be a supersingular elliptic curve over $\mathbb{F}_p$. Then, the group action can be efficiently evaluated for all the ideals $\mathfrak{l}_i$ lying above the ℓ_i via Vélu's formulae, as the kernel of the corresponding isogeny is defined over $\mathbb{F}_{p^2}$.

CSIDH simply instantiates the group action key exchange with the action of $\mathrm{Cl}(\mathcal{O})$ (restricted to the ideals lying above the ℓ_i) on the set of $\mathcal{O}$-oriented elliptic curves over $\mathbb{F}_p$, i.e. the supersingular ellptic curves defined over $\mathbb{F}_p$. Apart from being a reasonably performant key exchange which is conjectured to be post-quantum secure, public keys can be verified efficiently in CSIDH. As such, CSIDH was the first post-quantum proposal for a non-interactive (static-static) key exchange with reasonable performance. For the formalisation of the non-interactive key exchange protocol and multiple examples of real-world use cases of this protocol, we refer to [FHKP13].

Since its introduction, various works have further increased the performance of CSIDH. For example, one can instantiate CSIDH on the crater of the volcanoes by using curves oriented by the slightly larger order $\mathbb{Z}[\frac{1+\sqrt{-p}}{2}]$ for primes $p = 3 \bmod 4$ as suggested for CSIDH [CD20]. Another improvement is to use so-called *radical isogenies* to accelerate computing chains of d-isogenies in the CSIDH setting for small d. They allow us to avoid

generating points of order d for Vélu's formulae at each step and instead compute such points with explicit formulas deterministically [CDV20].

CSI-FiSh signatures. As was already sketched by Stolbunov in his thesis [Sto12], the class group action on oriented elliptic curves can be used to build a basic isogeny-based identification scheme. This identification scheme works as follows: The public key of the prover consists of an oriented curve $(E_1, \iota_1) = [\mathfrak{a}](E_0, \iota_0)$ for some random element $\mathfrak{a} \in \mathrm{Cl}(\mathcal{O})$ and (E_0, ι_0) a public starting curve. Now, assume $\mathrm{Cl}(\mathcal{O})$ is cyclic of order N and $\mathfrak{g}$ is a generator of $\mathrm{Cl}(\mathcal{O})$. To prove their identity, the prover first samples a random element $\mathfrak{b} \in \mathrm{Cl}(\mathcal{O})$, by choosing $b \in \mathbb{Z}/N\mathbb{Z}$ and setting $\mathfrak{b} = \mathfrak{g}^b$, and then commits to $[\mathfrak{b}](E_0, \iota_0)$. The verifier chooses a random bit $c \in \{0, 1\}$ and sends it to the prover, who replies with $r := b - c \cdot a \bmod N$. Finally, the verifier checks whether $[\mathfrak{g}^r](E_c, \iota_c)$ is equal to the commitment $[\mathfrak{b}](E_0, \iota_0)$. Using the Fiat–Shamir transform [FS87], the identification scheme can be turned into a signature scheme.

Conducting a record class group computation, Beullens, Kleinjung and Vercauteren computed the class group structure for the class group acting in CSIDH-512, the smallest of the CSIDH parameter sets. Using this data, they instantiated the Fiat–Shamir signature scheme sketched above for the class group action of CSIDH-512, which is called *CSI-FiSh*[2] [BKV19]. In Chapter 6, we will present a new isogeny-based group action which allows us to compute the class group structure more easily giving rise to digital signatures similar to CSI-FiSh for larger security levels.

2.3.2 SIDH

In 2011, Jao and De Feo introduced the *supersingular isogeny Diffie–Hellman* (SIDH) key exchange [JD11]. Predating CSIDH, this was the first proposal for an efficient isogeny-based key exchange at the time.

As a result of Proposition 2.2.39, we saw that for a fixed p the full supersingular ℓ-isogeny graph of elliptic curves defined over $\mathbb{F}_{p^2}$ is a connected Ramanujan graph for any $\ell \neq p$, where each ℓ-isogeny graph has the same set of vertices.

The idea behind the SIDH key exchange is for Alice and Bob to take a random walk starting from a publicly known curve E_0 in the full supersingular ℓ_1- and ℓ_2-isogeny graph, respectively. These walks correspond to isogenies of degree a power of ℓ_1 and ℓ_2, respectively. Alice then shares the codomain of her secret isogeny together with auxiliary information that allows Bob to compute the pushforward of his secret isogeny under

[2] pronounced "seafish"

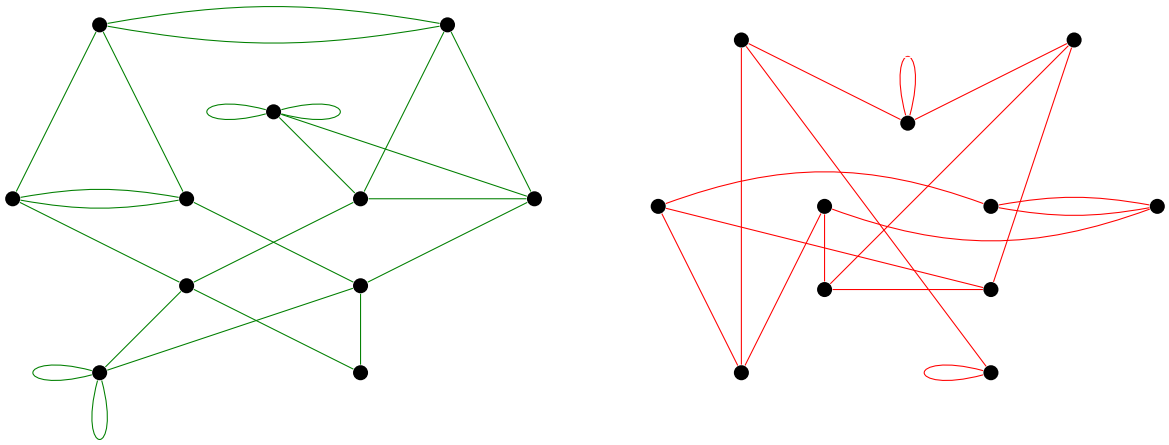

Fig. 2.4 The full supersingular 2- and 3-isogeny graphs for $\mathbb{F}_{127^2}$

Alice's isogeny. Bob proceeds analogously. After computing the pushforward of their respective secrets, both Alice and Bob arrive at isomorphic curves (see Fig. 2.2), whose j-invariant serves as the shared secret.

The following is a more detailed and more general description of the SIDH key exchange.

Public parameters. Let E_0 be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$, where p is a prime of the form $f \cdot N_1 N_2 \pm 1$. Here, N_1, N_2 are two coprime smooth integers, usually a power of 2 and 3, respectively, and f is a small cofactor. Furthermore, fix points P_A, Q_A, P_B, Q_B such that $E_0[N_1] = \langle P_A, Q_A \rangle$ and $E_0[N_2] = \langle P_B, Q_B \rangle$.

Key generation.

- Alice chooses her secret as a random cyclic subgroup of $E_0[N_1]$ generated by a point $A = P_A + [x_A]Q_A$ for some $x_A \in \mathbb{Z}/N_1\mathbb{Z}$. Similarly, Bob chooses his secret as $\langle B \rangle := \langle P_B + [x_B]Q_B \rangle \subset E_0[N_2]$ for some $x_B \in \mathbb{Z}/N_2\mathbb{Z}$.
- Alice computes the isogeny $\varphi_A : E_0 \to E_A := E_0/\langle A \rangle$ and sets her public key to be $((E_A, \varphi_A(P_B), \varphi_A(Q_B))$. Bob evaluates $\varphi_B : E_0 \to E_B := E_0/\langle B \rangle$ to compute his public key $(E_B, \varphi_B(P_A), \varphi_B(Q_A))$.

Key exchange.

- Both Alice and Bob can compute the j-invariant of $E_{AB} := E_0/\langle A, B \rangle$ as

$$E_{AB} \cong E_B/\langle \varphi_B(P_A) + [x_A]\varphi_B(Q_A) \rangle \cong E_A/\langle \varphi_A(P_B) + [x_B]\varphi_A(Q_B) \rangle.$$

Note that the choice of p ensures the torsion subgroups $E[N_1]$ and $E[N_2]$ to be defined over $\mathbb{F}_{p^2}$ for any supersingular elliptic curve over $\mathbb{F}_{p^2}$. Thus, one has an efficient kernel representation for all N_1- and N_2-isogenies. Since N_1 and N_2 are chosen to be smooth, these isogenies can be evaluated efficiently using Vélu's formulae. To ensure that both Alice and Bob enjoy the same level of security, the recommended parameter sets for SIDH suggest *balanced* parameters, in other words $N_1 \approx N_2$.

In spite of its name, SIDH lacks certain features of the classical Diffie–Hellman key exchange. For instance, the scheme does not come with an easy method to validate public keys. By sending manipulated torsion point images in their public key, adversaries can trick the other party into revealing information about their secret. In Section 3.2, we give more details about this adaptive attack due to Galbraith, Petit, Shani and Ti [GPST16].

SIKE. SIDH is the basis of the only isogeny-based encryption scheme, called SIKE, submitted to NIST's post-quantum standardisation process started in 2017 [JAC+17]. To avoid adaptive attacks, a variant of the Fujisaki-Okamoto transformation [FO99] due to Hofheinz, Hövelmanns and Kiltz [HHK17] is applied to SIDH resulting in SIKE.

B-SIDH. Choosing the prime p of the form $N_1N_2f \pm 1$ with $N_1 \approx N_2$ implies that the curves E_A, E_B are connected to E_0 by an isogeny of degree roughly $\sqrt{p}$. However, the diameter of the full ℓ-isogeny graph is larger than $\log_\ell(\sqrt{p})$, where each step corresponds to a degree ℓ-isogeny. In other words, two randomly chosen supersingular elliptic curves over $\mathbb{F}_{\overline{p}}$ in a full supersingular ℓ-isogeny graph will usually not be connected by an isogeny of degree roughly $\sqrt{p}$.

In order to avoid walking only in a "small" subgraph (relative to the size of the full isogeny graph) and to reduce the size of the prime p, Costello introduced the variant B-SIDH [Cos20]. The main differences between SIDH and B-SIDH are

- N_1 and N_2 have to be divisors of $p-1$ and $p+1$, respectively. Hence, $p-1$ and $p+1$ both need to have large smooth factors as opposed to just one of them.
- One has $N_1 \approx N_2 \approx p$ as opposed to $N_1 \approx N_2 \approx \sqrt{p}$ in SIDH.
- Kernel generators are a priori $\mathbb{F}_{p^4}$-rational as opposed to $\mathbb{F}_{p^2}$-rational.

In B-SIDH, the curves E_0 and E_A are no longer closer than expected in the isogeny graph, but parameter selection might be harder and it seems at first to come at the expense of working over larger field extensions. However, for every supersingular elliptic curve E defined over $\mathbb{F}_{p^2}$, there exists a quadratic twist. If E has $(p+1)^2$ rational points over $\mathbb{F}_{p^2}$,

then its twist has $(p-1)^2$ rational points over $\mathbb{F}_{p^2}$. Thus, when computing an isogeny of degree N_1 dividing $p+1$ one can work with the curves having $p+1$ rational points, and before computing an isogeny of degree N_2 dividing $p-1$, one switches to twists that have $p-1$ rational points. Technically, the switch makes it possible to compute the isogenies using only operations over $\mathbb{F}_{p^2}$. For more details, we refer to [Cos20].

Attacks on SIDH. Unfortunately, the auxiliary information exchanged between Alice and Bob to compute the pushforward of their isogenies turned out to be the Achilles heel of SIDH leading to the downfall of the scheme in 2022. A series of recent attacks breaks SIDH (and B-SIDH) efficiently, exploiting the torsion point images published in the SIDH protocol [CD22, MM22, Rob22a]. We will discuss these attacks as well as the previous so-called *torsion point attacks* in more detail in Chapter 3. Countermeasures have been proposed to prevent the recent attacks, e.g. masking the degree of the secret isogeny [Mor22] or the torsion point information [Fou22]. However, the resulting schemes are less efficient and less practical than SIDH.

2.4 Problems underlying isogeny-based cryptography

In this section, we will briefly introduce some computational problems appearing in isogeny-based cryptography, highlight several reductions between these problems and we recall the complexity of solving these problems given algorithms known at present.

We start with the problem at the core of isogeny-based cryptography.

Definition 2.4.1. Let $E/\mathbb{F}_q, E'/\mathbb{F}_q$ be two isogenous elliptic curves. The *pure isogeny problem* asks to compute an isogeny between E and E'.

Usually, we implicitly require solutions to the pure isogeny problem to have an efficient representation and to be able to evaluate the isogeny.

Computing isogenies between isogenous ordinary elliptic curves. The fastest algorithms known to compute an isogeny between ordinary elliptic curves are variants of an algorithm due to Galbraith [Gal99]. Recall from Example 2.2.27 that every ordinary elliptic curve over $\mathbb{F}_q$ comes with a natural orientation induced by the q-power Frobenius, say by $\mathbb{Z}[\sigma]$. Let K be the imaginary quadratic field containing $\mathbb{Z}[\sigma]$ and let $\mathcal{O}_K$ denote its ring of integers. The idea of Galbraith's algorithm is to consider every ℓ-isogeny volcano for ℓ dividing the conductor of $\mathbb{Z}[\pi]$ in $\mathcal{O}_K$ and to walk to the crater of every volcano from both given curves. The two resulting curves (with isomorphic endomorphism rings)

are connected by a horizontal isogeny. Computing this horizontal isogeny is the most expensive step asymptotically. Classically, one finds the horizontal isogeny in exponential time $O^*(q^{1/4})$ using a meet-in-the-middle approach, i.e. by growing trees of random isogenies from both curves until they collide. On a quantum computer, the isogeny can be computed in subexponential time by reducing the problem of finding the isogeny to a hidden shift problem [CJS14].

If enough information is provided to walk to the craters, the same idea can be applied for oriented elliptic curves in general. Note that one could also apply the reduction to the hidden shift problem directly whenever two curves are oriented by the same primitive order.

Computing isogenies between isogenous supersingular elliptic curves. We have seen that the full supersingular ℓ-isogeny graph is connected. Using Proposition 2.2.10, one can see that pathfinding in this graph using a meet-in-the-middle approach takes $O^*(p^{1/2})$ time and memory. The same time complexity but requiring significantly less memory is achieved by an algorithm due to Delfs and Galbraith [DG16]. The idea behind the algorithm is to split the isogeny computation into parts. First, one uses random walks to connect the given supersingular elliptic curves to the subgraph of supersingular elliptic curves (isomorphic to one) defined over $\mathbb{F}_p$, which is easily recognisable as the j-invariant lies in $\mathbb{F}_p$. Within this subgraph containing roughly $\sqrt{p}$ vertices, the remaining isogeny can be computed using a meet-in-the-middle approach. Using a quantum computer, the first step of the algorithm by Delfs and Galbraith can be accelerated using Grover's search as shown by Biasse, Jao and Sankar, reducing the complexity of the algorithm to $O^*(p^{1/4})$ [BJS14]. Further, Tani's quantum claw finding algorithm [Tan09] is claimed to recover isogenies of smooth given degree faster than a classical meet-in-the-middle approach. However, modelling the costs for accessing memory in Tani's algorithm differently leads to divergent complexity claims [JS19].

To recover horizontal isogenies between oriented supersingular elliptic curves such as in the case of CSIDH, the subexponential quantum attack by Childs, Jao and Soukharev also applies. Yet, the concrete complexity of this attack has been a matter of debate. To solve the hidden shift problem, variants of Kuperberg's collimation sieve are used [Kup05]. Depending on the type of memory considered, different cost estimates have been suggested [Pei20, BS20, CSCJR22].

Computing endomorphism rings. As noted before, the endomorphism ring of an elliptic curve carries a lot of information about the curve which makes the following problem a natural one to consider.

Definition 2.4.2. Let $E/\mathbb{F}_q$ be an elliptic curve. The *endomorphism ring computation problem* asks to compute $\mathrm{End}(E)$.

In the ordinary case, the endomorphism ring of an elliptic curve can be computed classically in heuristic subexponential time [BS11, Bis12] and efficiently on quantum computers [Rob22b].

The supersingular case is more interesting to us. It turns out that the problem of solving the pure isogeny problem for supersingular elliptic curves is equivalent to computing their endomorphism rings. This was first shown assuming some heuristics, see for example [EHL+18], and later proven to be true by Wesolowski assuming only the Generalised Riemann Hypothesis (GRH) [Wes22b].

The problem of computing endomorphism rings of supersingular elliptic curves was first studied by Kohel in his thesis [Koh96]. To compute endomorphisms, loops in the full supersingular ℓ-isogeny graph containing the curve are searched for some ℓ, which is a special pathfinding problem. In particular, one can also use the algorithm by Delfs and Galbraith to compute loops or its quantum version due to Biasse, Jao and Sankar [DG16, BJS14]. The fastest classical algorithm known to date to compute endomorphism rings of supersingular elliptic curves is due to Eisenträger, Hallgren, Leonardi, Morrison and Park [EHL+20]. It uses a more direct approach to compute the endomorphism ring and does not require to find isogeny paths, running heuristically in time $O(\log(p)^2 p^{1/2})$ with polynomial storage requirements.

Additional constraints. For practical schemes in cryptography, usually variants of the pure isogeny problem are used. To recover a secret one may, for instance, be required to find an isogeny of a given degree or an isogeny which has a prescribed action on certain points between two given elliptic curves. A priori it is not clear whether fixing additional constraints would make the pure isogeny problem harder or easier to solve, as it provides additional information and simultaneously reduces the number of possible solutions.

Clearly, computing an unknown isogeny of specific degree d between two d-isogenous elliptic curves can always be done using an exhaustive search over all $O(d)$ isogenies of degree d (or equivalently their kernels). If d is a prime, this is in fact the best currently known method.

When d is a smooth integer a meet-in-the-middle approach with $O^*(\sqrt{d})$ time and memory complexity can be used. If d is large, the memory requirement becomes unrealistic.

Bounding the available memory leads to the conclusion that a van Oorschot–Wiener collision search whose concrete complexity depends on the amount of memory available is more efficient to compute the isogeny in this case [CLN+20].

Note that computing the endomorphism rings of two d-isogenous supersingular elliptic curves and then using the previously mentioned reduction by Wesolowski [Wes22b] to compute a connecting isogeny will in general not return an isogeny of a specific degree. However, when the endomorphism rings of the two supersingular curves are known (or they have been precomputed), d does not need to be smooth but merely the product of two factors of roughly the same size to make a meet-in-the-middle approach work to compute an unknown isogeny of degree d. This is because isogenies corresponding to factors of d too large to be computed using Vélu's formulae can be replaced by an isogeny of powersmooth degree using for instance the KLPT algorithm [KLPT14] briefly sketched in Section 5.2.1. While replacing the large degree isogenies adds to the overhead of the meet-in-the-middle approach, the isogenies of powersmooth degree can still be computed in order to find a collision.

In Chapter 5, we give a reduction of the problem of recovering an isogeny of a *specific degree* d between two given curves to the problem of computing their endomorphism ring, if images of a group of roughly size d are known under the unknown isogeny.

The case of a fixed known degree and given torsion point images under the sought isogeny is the setting which appears in SIDH and its variations. Due to its central importance for this thesis, we state this variation of the pure isogeny problem explicitly as follows.

Definition 2.4.3 (Supersingular Isogeny with Torsion (SSI-T))**.** Let p be a prime and N_1, N_2 be smooth, coprime integers. Given two supersingular elliptic curves E_0 and E_A over $\mathbb{F}_{p^2}$ connected by an isogeny $\varphi_A : E_0 \to E_A$ of known degree N_1 and given the restriction of φ_A to $E_0[N_2]$, the *SSI-T problem* asks to find an isogeny $\varphi : E_0 \to E_A$ matching these constraints.

It was first shown by Petit that the restriction of an isogeny of known degree N_1 with $N_1^4 > p$ to a sufficiently large subgroup of size N_2 allows to retrieve such an isogeny [Pet17]. More recently, the required size of the subgroup was significantly reduced, breaking the security of the SIDH key exchange [CD22, MM22, Rob22a]. We discuss these so-called *torsion point attacks* in more detail in Chapter 3.

CHAPTER 3

SIDH Attacks Using Torsion Point Images

This chapter contains an improvement of torsion point attacks as first introduced by Petit [Pet17, dQKL+21]. This improvement is unpublished joint work with Mingje Chen, Péter Kutas, Christophe Petit and Yiming Tang. In Section 3.5, we present a reduction from the problem underlying overstretched and imbalanced SIDH to an abelian hidden shift problem previously published as

Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Charlotte Weitkämper. One-way functions and malleability oracles: Hidden shift attacks on isogeny-based protocols. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 242–271. Springer, Heidelberg, October 2021.

For the sake of completeness, we finish the chapter with a brief sketch of the recent devastating attacks against SIDH published by Castryck and Decru [CD22], Maino and Martindale [MM22], and Robert [Rob22a] for which we do not claim any contributions.

3.1 Introduction

As modelled by the SSI-T problem, a malicious party trying to attack SIDH not only gets two isogenous elliptic curves but also some torsion point images under the sought isogeny and its degree. Originally, it was not clear whether this additional information could be exploited to weaken the scheme. The first work exploiting this information for an active attack was by Galbraith, Petit, Shani and Ti, using manipulated torsion point information to iteratively recover *static* secret SIDH keys [GPST16]. We briefly recall their attack in Section 3.2.

Another line of work shows that the provided torsion point information together with the knowledge of the endomorphism ring of the starting curve can be used to mount a passive attack on non-standard variants of SIDH. We use the notation from Section 2.3.2, where we introduced the SIDH key exchange. Recall that the integers N_1 and N_2 are the degrees of the secret isogenies of Alice and Bob, respectively, and p is the characteristic of the finite field that the supersingular elliptic curves are defined over. Petit showed that the secret isogeny of Alice can be recovered in classical polynomial time, whenever $N_2 > N_1^4 > p$ [Pet17]. However, SIDH and other SIDH-based schemes proposed in the literature so far, for instance B-SIDH [Cos20], take N_1 and N_2 to be divisors of $p^2 \pm 1$. This ensures that torsion points used in the schemes are defined over small field extensions, making the resulting schemes more efficient. Thus, torsion point attacks with *overstretched parameters*, i.e. $N_1, N_2 > p$, were often dismissed as unrealistic.

At Crypto 2021, improvements to the torsion point attacks were presented [dQKL+21]. The authors managed to extend Petit's torsion point attacks to more practical parameter sets where N_1 and N_2 are divisors of $p \pm 1$ or $p^2 \pm 1$. They proposed two new attack variants: the *dual isogeny attack* and the *Frobenius isogeny attack*. The dual isogeny attack runs in polynomial time whenever the starting curve has j-invariant 1728 and $N_2 > pN_1$. The Frobenius isogeny attack also requires a special starting curve with a known endomorphism ring and runs in polynomial time whenever $N_2 > \sqrt{p}N_1^2$. These

results had an impact on imbalanced versions of SIDH and B-SIDH, where we do not have $N_1 \approx N_2$. One would get such parameter sets for example by extending SIDH to a group key exchange in the most natural way. Finally, exponential time attacks that are faster than the generic meet-in-the-middle algorithm are discussed in [dQKL+21]. Their main conclusion was that for SIDH-like parameters one can do better than meet-in-the-middle algorithms whenever $N_2 > N_1^3$. In Section 3.3, we recall the details behind these torsion point attacks. Then, we present improvements that allow us to further reduce the required imbalance of parameters in Section 3.4. More precisely, we show how precomputation may be used to do better than meet-in-the-middle algorithms also for $N_2 > N_1^{2.5}$. Our improvements to the attack stem from finding "better" solutions to a norm equation in the quaternion algebra.

Considering certain SIDH variants with overstretched and imbalanced parameter sets also allows us to give further reductions between hardness assumptions. In Section 3.5, we provide a new quantum attack for SIDH variants using such parameters by reducing the underlying computational problem to an injective abelian hidden shift problem. This can be solved in quantum subexponential time by algorithms such as Kuperberg's [Kup05, Kup11]. While only valid for parameters that were already broken classically by aforementioned torsion point attacks, this reduction disproved a widespread belief amongst cryptographers that due to SIDH's non-commutative nature, no reduction to an abelian hidden shift problem would be possible. In particular, many believed that no reasonable variant of Childs–Jao–Soukharev's attack [CJS14] applies in the case of SIDH [JD11, p. 18, Sect. 5].

Finally, attacks can only get better. In summer of 2022, a series of attacks by Castryck and Decru [CD22], Maino and Martindale [MM22], and Robert [Rob22a] were published that exploit the torsion point information provided to break (balanced) SIDH and various related schemes completely. We briefly summarise these recent attacks and their implications in Section 3.6.

3.2 Active GPST attack on semi-static SIDH

We briefly recall the adaptive attack due to Galbraith, Petit, Shani and Ti [GPST16]. For the exposition, we further assume that the secret isogenies of Alice and Bob are of degree $N_1 = 2^{e_A}$ and $N_2 = 3^{e_B}$, respectively, as was typically suggested. The adaptive GPST attack actively recovers a *static* SIDH key x_A of a party, say Alice, where $\langle P_A + [x_A]Q_A \rangle$ is the subgroup corresponding to the kernel of her secret isogeny. The attacker uses the

success of a key exchange as the following oracle which is used to recover Alice's static key bit-wise.

Definition 3.2.1 (Oracle in static SIDH)**.** Upon receipt of an elliptic curve E_0, two linearly independent points $R, S \in E_0[2^{e_A}]$ of order 2^{e_A} and another elliptic curve E', the oracle responds 1 if $j(E_0/\langle R + [x_A]S\rangle) = j(E')$ and 0 otherwise.

To recover Alice's secret key, an attacker first generates their public key honestly as $(E_B, R := \varphi_B(P_A), S := \varphi_B(Q_A))$ as specified by the SIDH key exchange. Then, they query the oracle on $(E_B, R, S + [2^{e_A-1}]R, E_{AB})$, which reveals whether the elliptic curves $E_B/\langle R + [x_A](S + [2^{e_A-1}]R)\rangle$ and $E_B/\langle R + [x_A]S\rangle$ are isomorphic. By the following lemma, this reveals the least significant bit of the static secret x_A.

Lemma 3.2.2. *[GPST16, Lem. 2] For linearly independent $R, S \in E[2^{e_A}]$ of order 2^{e_A}, x_A is even if and only if $\langle R + [x_A](S + [2^{e_A-1}]R)\rangle = \langle R + [x_A]S\rangle$.*

Afterwards, the attacker can proceed iteratively to recover further bits. Let K_i denote the partial key corresponding to the least significant i bits of x_A, i.e. $K_i = x_A \bmod 2^i$, and assume the attacker has recovered K_i. To learn the next bit of x_A, the attacker simply queries the oracle on

$$\Big(E_B, (R - [2^{e_A-i-1}][K_i]S), ([1 + 2^{e_A-i-1}]S), E_{AB}\Big). \tag{3.1}$$

The next bits of x_A can be deduced from the oracle's answer using the following lemma, which concludes the active attack by Galbraith, Petit, Shani and Ti.

Lemma 3.2.3. *Given the query* (3.1)*, the oracle in static SIDH (Definition 3.2.1) returns* 1 *if and only if the $(i+1)$-th least significant bit of x_A is* 0*.*

Proof. The curve computed by Alice is E_B/G' where

$$\begin{aligned} G' = \langle R' + [x_A]S'\rangle &= \langle (R - [2^{e_A-i-1}][K_i]S) + [x_A]([1 + 2^{e_A-i-1}]S)\rangle \\ &= \langle R + [x_A]S + [x_A - K_i][2^{e_A-i-1}]S)\rangle. \end{aligned}$$

This equals $\langle R + [x_A]S\rangle$ if and only if the $(i+1)$-th bit of x_A is 0. □

Remark 3.2.4. Using the exact queries as described in this section, the attack can be detected by Alice using the Weil pairing. However, as described in [GPST16], an attacker might choose a suitable scalar to scale both of the malicious torsion points in order to evade this detection. This can be done for all but the two most significant bits for which there would not be a suitable scalar, but that can be brute-forced easily. For further details about the validation and the scaling, we refer to [GPST16].

3.3 Classical torsion point attacks

Next, we briefly survey the so-called *torsion point attacks* due to Petit [Pet17] and the later improvements from [dQKL+21]. These results target to solve the SSI-T problem for certain parameters N_1 and N_2, given the endomorphism ring of the starting curve $E_0/\mathbb{F}_{p^2}$. In the following, we will always assume that N_1, N_2 are smooth integers, the prime $p \equiv 3 \bmod 4$ and that the N_1N_2-torsion of the elliptic curves involved can be efficiently represented, as is the case for instance with SIDH parameters.

3.3.1 Endomorphisms for classical torsion point attacks

It is easy to see that the existence of one isogeny $E_0 \to E_A$, e.g. the sought isogeny, implies the embedding of a suborder of $\mathrm{End}(E_0)$ in $\mathrm{End}(E_A)$. The core tool behind torsion point attacks is to use the provided torsion point information in the SSI-T problem to compute one endomorphism of this embedding, depicted on the left of E_A in Fig. 3.1 and sometimes referred to as a *lollipop endomorphism*.

The following Theorem explains why recovering one such endomorphism can help to compute the secret isogeny in an SIDH variant [dQKL+21, Thm. 3].

Theorem 3.3.1 ([dQKL+21])**.** *Let $\varphi : E_0 \to E_A$ be a secret isogeny of degree N_1 between two supersingular elliptic curves over $\mathbb{F}_{p^2}$, where N_1 has $\log\log p$ distinct prime factors. Assume that $E[N_1]$ and $E[N_2]$ are efficiently representable for any supersingular curve E and that the action of φ on $E_0[N_2]$ is given. Further, suppose the restriction of a trace-zero endomorphism $\theta \in \mathrm{End}(E_0)$ to $E_0[N_2]$, an integer d coprime to N_2, and a smooth integer e such that*

$$\deg(\varphi \circ \theta \circ \widehat{\varphi} + [d]) = N_2^2 e.$$

are given. Then, we can compute φ in time $O^*(\sqrt{e}) = O(\sqrt{e} \cdot \mathrm{poly}(\log(p)))$.

Proof. We briefly recall the main ideas of the proof here. Let $\tau = \varphi \circ \theta \circ \widehat{\varphi} + [d]$. If $\ker(\tau)$ is cyclic, then $\tau = \widehat{\psi'} \circ \eta \circ \psi$, where $\deg(\psi) = \deg\left(\widehat{\psi'}\right) = N_2$, $\deg(\eta) = e$ and the kernels of ψ and $\widehat{\psi'}$ are both cyclic. See Fig. 3.1, for a depiction of the two decompositions. In [dQKL+21], it is shown that $\ker(\tau)$ is always cyclic if N_2 is odd and if N_2 is even, then $\tau = \widehat{\psi'} \circ \eta \circ \psi \circ [m]$, where $\deg(\psi) = \deg\left(\widehat{\psi'}\right) = N/m$, $\deg(\eta) = e$ and $m = 1$ or $m = 2$.

Now ψ and m can be computed using the torsion point information and $\widehat{\psi'}$ using the observation that $\ker\left(\widehat{\psi'}\right) = \tau\left(E_A[N_2]\right)$. The isogeny η can be computed by a meet-in-the-middle algorithm in time $O^*(\sqrt{e})$. Once τ is known, one can determine φ by looking

at $G = \ker(\varphi \circ \theta \circ \hat{\varphi}) \cap E_A[N_1]$. If G is cyclic, it can be recomputed easily. If not, then τ can be recovered using [Pet17, Sect. 4.3]. This last search is efficient by the condition that N_1 has $O(\log \log p)$ distinct prime factors. □

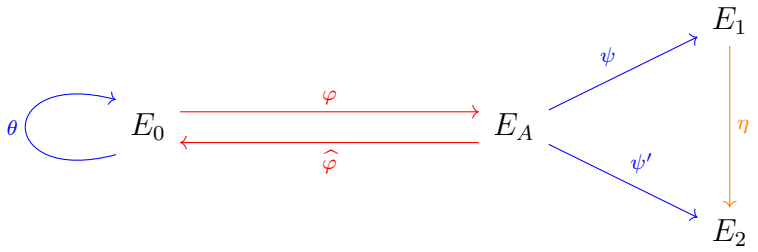

Fig. 3.1 Decomposing the (shifted) *lollipop endomorphism* $\varphi \circ \theta \circ \widehat{\varphi} + [d]$ of degree $N_2^2 e$ as $\psi \circ \eta \circ \widehat{\psi'}$, where $\deg(\psi) = \deg(\psi') = N_2$ and $\deg(\eta) = e$ as in Theorem 3.3.1.

Remark 3.3.2. More generally, let M be the largest divisor of N_1 such that $E_A[M]$ is contained in $\ker(\varphi \circ \theta \circ \hat{\varphi}) \cap E_A[N_1]$ and let k be the number of distinct prime factors of M. Then the last step in the proof to recover τ using [Pet17, Sect. 4.3] takes time $O^*(2^k)$. Thus, when dropping the condition on N_1's number of distinct prime factors, the algorithm still works with a complexity of $O^*(2^k\sqrt{e})$. Further, it can be easily verified that the same theorem and proof carry over when the action of φ on $E_0[N_2]$ is only known up to a fixed but unknown scalar multiple coprime to N_2.

Theorem 3.3.1 implies that one obtains an attack to solve the SSI-T problem improving upon meet-in-the-middle isogeny search of the secret isogeny if one can find a suitable e that is sufficiently small (and smooth). More precisely, one first tries to recover suitable θ, d and e before invoking Theorem 3.3.1. In [dQKL+21] this attack was called the *dual isogeny attack*. The paper further provided an alternative strategy, the *Frobenius attack*, which requires slightly different θ, d and e and uses the Frobenius isogeny as follows.

Theorem 3.3.3 ([dQKL+21])**.** *Let $\varphi : E_0 \to E_A$ be a secret isogeny of degree N_1 between two supersingular elliptic curves over $\mathbb{F}_{p^2}$, where N_1 has $\log \log p$ distinct prime factors. Assume that $E[N_1]$ and $E[N_2]$ are efficiently representable for any supersingular curve E and that the action of φ on $E_0[N_2]$ is given. Further, suppose the restriction of a trace-zero endomorphism $\theta \in \mathrm{End}\,(E_0)$ to $E_0[N_2]$, an integer d coprime to N_2, and a smooth integer e such that*

$$\deg\left(\varphi \circ \theta \circ \widehat{\varphi} + [d]\right) = N_2^2 p e.$$

are given. Then, we can compute φ in time $O^*\left(\sqrt{e}\right) = O(\sqrt{e} \cdot \mathrm{poly}(\log(p)))$.

Proof. Let $\tau = \varphi \circ \theta \circ \hat{\varphi} + [d]$. As in the proof of Theorem 3.3.1, τ can be decomposed as $\widehat{\psi'} \circ \eta \circ \psi \circ [m]$, where ψ of degree N_2 and m can be computed efficiently and η is of degree pe. To recover η, we observe that η has inseparable degree p as we are in the supersingular case. Let π denote the Frobenius isogeny, then $\eta = \pi \circ \eta'$ and η' can be recovered using a generic meet-in-the-middle algorithm as before. □

Theorems 3.3.1 and 3.3.3 provide a reduction from the SSI-T problem to the following problem with $B = N_2^2$ and $B = N_2^2 p$ respectively.

Problem 3.3.4. Let p be a prime, N_1 and N_2 be smooth coprime integers, and $E_0/\mathbb{F}_{p^2}$ a supersingular elliptic curve. Given a positive integer B, find the restriction of a trace-zero endomorphism $\theta \in \mathrm{End}(E_0)$ to $E_0[N_2]$, an integer d coprime to N_2, and a smooth (small) integer $0 < e$ such that

$$\deg(\varphi \circ \theta \circ \widehat{\varphi} + [d]) = N_1^2 \deg(\theta) + d^2 = Be.$$

Note, that the above problem only depends on (p, N_1, N_2, E_0) and not on any unknown isogeny specific to a particular instance of the SSI-T problem. Thus, if this problem is solved, arbitrary instances of SIDH with these fixed parameters could be solved.

3.3.2 Solving norm equations

In this subsection, we recall the parameters given by [dQKL+21] under which a solution to Problem 3.3.4 can be computed when the starting curve is $E_0 : y^2 = x^3 + x$ with j-invariant 1728. Recall from Example 2.2.25 that every endomorphism of E_0 is an integer linear combination of $1, \iota, (\iota + \pi)/2, (1 + \iota\pi)/2$. For simplicity, we drop the denominators and we are looking for a suitable trace-zero endomorphism θ as a linear combination of ι, π and $\iota\pi$. That is, there exist $a, b, c \in \mathbb{Z}$ such that $\theta = a\iota\pi + b\pi + c\iota$, so $\mathrm{Norm}(\theta) = N_1^2 (pa^2 + pb^2 + c^2)$. Problem 3.3.4 thus motivates solving the Diophantine equation

$$N_1^2 \left(pa^2 + pb^2 + c^2\right) + d^2 = Be, \tag{3.2}$$

where one is looking for integers $a, b, c, d \in \mathbb{Z}$ and a small and smooth positive integer e.

A strategy to solve Eq. (3.2) in the case of the dual isogeny attack, where $B = N_2^2$, was described in [Pet17]. First, one considers the equation modulo N_1^2 assigning an appropriate value to d. Then, one is left with solving $pa^2 + pb^2 + c^2 = \frac{N_2 e - d^2}{N_1^2}$ which can be accomplished by solving modulo p using Cornacchia's algorithm, if $\left(\frac{Be-d^2}{A^2} - c^2\right) p^{-1}$ is a sum of two squares. If this is not the case, then a new value for c is chosen.

This method can be shown to work essentially whenever $N_2 > pN_1$ for $B = N_2^2$ and general e [dQKL+21].

Finally, the Frobenius attack by [dQKL+21] leads to the equation

$$N_1^2\left(pa^2 + pb^2 + c^2\right) + d^2 = N_2^2 pe,$$

i.e. with $B = N_2^2 p$. For this equation, one has to choose c, d to be divisible by p as there would not be a solution modulo p otherwise. By invoking this change of variables and dividing by p we obtain the following new equation

$$N_1^2\left(a^2 + b^2 + pc^2\right) + pd^2 = N_2^2 e. \tag{3.3}$$

Similarily to before, one sets e to be a value such that the above equation is solvable modulo N_1^2. Then, one solves modulo N_1^2. For simplicity, set $c = 0$ (which is only necessary when $\frac{N_2^2 e - pd^2}{N_1^2} \ll p$) and check whether $\frac{N_2^2 e - pd^2}{N_1^2}$ is a sum of two squares. If not, then one chooses a different d. This method was shown to work whenever $N_2 > \sqrt{p}N_1^2$ [dQKL+21].

Note that the two improvements work for different types of parameters. However, when studying imbalanced but not overstretched SIDH variants, i.e. where $N_1 N_2 \approx p$, the Frobenius attack is the one to choose. In this case, the method leads to a polynomial time attack on SIDH-like schemes whenever $N_2 > N_1^5$.

Instead of only considering attacks running in polynomial time, one can also consider attacks which run asymptotically worse but that still perform better than generic meet-in-the-middle attacks. This raises the question how balanced the parameters N_1 and N_2 can be for the known torsion point attacks to run with a time complexity lower than $O^*(\sqrt{N_1})$. There are two natural approaches to this question: increasing e and recovering a larger isogeny in the end of torsion point attacks or guessing part of the secret isogeny. Both approaches are analysed in [dQKL+21]. The main conclusion of their analysis is that for SIDH-like parameters with $N_1 N_2 \approx p$ one gets a classical attack outperforming the meet-in-the-middle search whenever $N_2 > N_1^3$.

3.4 Improving torsion point attacks by using precomputation

Next, we sketch an unpublished idea predating the devastating attacks on SIDH presented in Section 3.6. This idea was jointly worked on with Mingje Chen, Péter Kutas, Christophe Petit and Yiming Tang. We improve the Frobenius isogeny attack by [dQKL+21], recalled

in the previous section, by extending it to slightly less imbalanced parameter sets. This is achieved by a modification in the strategy to find solutions (a, b, c, d, e) to Eq. (3.3):

$$N_1^2 \left(a^2 + b^2 + pc^2\right) + pd^2 = N_2^2 e.$$

Recall that the cost of the torsion point attack depends on the size (and smoothness) of e, as it ultimately requires from the attacker to find an isogeny of degree e. In particular, we want e to be smaller than N_1 and as smooth as possible to improve upon a generic meet-in-the-middle attack. By looking for e of the form $e_0 e_1^2$, one can turn the problem of finding a solution to the norm equation into a problem of finding a lattice vector in a centrally symmetric rectangle as we will show in the following.

For a fixed e_0 this view does not lead to asymptotic improvements over previous attacks. However, if we loop through many choices for e_0 and therefore different lattices, we can find lattices with particularly short vectors, i.e. shorter than the ones guaranteed to exist by Minkowski's theorem for every single lattice. Under a few heuristics and using techniques from [AEN19], we sketch an analysis for our approach.

The upshot is to iterate through lattices until we find one with a particularly short vector in a precomputation. This lowers the required imbalance of parameters for the torsion point attacks. Using this refinement of the attack, the torsion point attacks are still faster than meet-in-the-middle algorithms when $N_2 > N_1^{2.5}$ using a precomputation of time $O^*(\sqrt{N_1})$. This should be compared to the attack of the previous section which outperforms meet-in-the-middle algorithms whenever $N_2 > N_1^3$ without any precomputation.

3.4.1 Algorithm

We are interested in torsion point attacks where $N_1 N_2 \approx p$ and parameters are imbalanced. In particular, we are interested in $N_1^3 > N_2 > N_1^{2.5}$ as this will keep the (offline) precomputation in the following lower than the cost of a direct attack. As before, we aim to find integer solutions a, b, c, d to Eq. (3.3) with $e \in \mathbb{Z}$ as small and smooth as possible. Once a solution is found, the complexity of the resulting torsion point attack depends on the complexity of recovering an isogeny of degree e using a meet-in-the-middle algorithm as was shown in Theorem 3.3.3.

First, we look at a relaxed version of Eq. (3.3), where we substitute $f := a^2 + b^2 + pc^2$.

$$N_1^2 f + pd^2 = N_2^2 e. \tag{3.4}$$

Note, that this simplification of the equation does not hide much information, since for our imbalanced parameters ($N_2 > N_1^{2.5}$ and $N_1 N_2 \approx p$) the integer f is significantly larger than p, as $f \approx N_2^2 e / N_1^2$ and $e < N_1$. In this case, most integers f are of the form $a^2 + b^2 + pc^2$ and this representation can be computed efficiently. To see this, one can guess values for c until $f - pc^2$ is a prime congruent to 1 mod 4 and then compute the decomposition of into the sum of two squares using Cornacchia's algorithm [Cor08].

Considering Eq. (3.4) modulo N_1^2, we get $N_2^2 e - pd^2 \equiv 0 \pmod{N_1^2}$. We consider e of the form $e_0 e_1^2$, with e_0 chosen such that a solution to the congruence $N_2^2 e - pd^2 \equiv 0 \pmod{N_1^2}$ exists. Denote by τ_0 a square root of $N_2^2 e_0 p^{-1}$. Then $(d, e_1) = (\tau_0, 1)$ is a solution to the congruence

$$N_2^2 e_0 e_1^2 - pd^2 \equiv 0 \bmod N_1^2. \tag{3.5}$$

We consider the integer lattice $\mathcal{L}$ generated by $(\tau_0, 1), (N_1^2, 0)$ in which every vector corresponds to a solution for (d, e_1) to Eq. (3.5).

Since f is a sum of squares, it needs to be positive, giving a condition on d. Further, for the torsion point attacks we want to find a vector in this lattice whose second coordinate, corresponding to e_1, is sufficiently small. Both conditions yield the inequalities

$$N_2^2 e_0 e_1^2 - pd^2 > 0, \quad \sqrt{e_0} e_1 < \sqrt{N_1}, \tag{3.6}$$

which can be rewritten as

$$d < \frac{N_2 \sqrt{e_0} e_1}{\sqrt{p}}, \quad e_1 < \frac{\sqrt{N_1}}{\sqrt{e_0}}. \tag{3.7}$$

The bounds also imply that $d < \frac{N_2 \sqrt{N_1}}{\sqrt{p}}$. In summary, we are left looking for a vector (d, e_1) in $\mathcal{L}$ such that $e_1 < \frac{\sqrt{N_1}}{\sqrt{e_0}}$ and $d < \frac{N_2 \sqrt{N_1}}{\sqrt{p}}$. If such vectors exist, they can be found by a simple and efficient Lagrange–Gauss reduction using a weighted norm (or alternatively by rescaling the lattice to $\mathcal{L}_{e_0}$ generated by $(\sqrt{p} N_1^2, 0)$ and $(\sqrt{p}\tau_0, N_2 \sqrt{e_0})$ and just using the standard euclidean norm).

A natural way of proving the existence of short vectors is by applying Minkowski's theorem. The bounds imply that a suitable vector is inside a rectangle centred around the origin as we only care about e_1^2 and d^2 bounded by the two inequalities. The area of the rectangle is $\frac{N_1 N_2}{\sqrt{e_0 p}}$ and the determinant of the lattice $\mathcal{L}$ is N_1^2. This implies heuristically that once $N_1 N_2 > C \sqrt{e_0 p} N_1^2$ for some constant C, a suitable lattice vector should exist. Unfortunately, some of these vectors will not lead to solutions of the original equation due to the loss of information when combining the multiple inequalities. Yet, with several

solutions available one heuristically expects to find a suitable one, which we confirmed experimentally.

For a fixed e_0 the bound given above does not improve upon previous state-of-the-art torsion point attacks from [dQKL+21]. However, our idea is to instead iterate through many different choices for e_0. In each step, we fix an e_0 and look at the resulting lattice $\mathcal{L}_{e_0}$. For each of them, Minkowski's theorem provides a bound for the shortest vector. However, Minkowski's theorem is not an equivalence statement and lattices with shorter vectors than predicted by Minkowski's theorem exist.

In Section 3.4.2, we sketch an analysis determining how many random lattices of a fixed determinant need to be sampled to find vectors of unexpectedly short length that will suffice for our purpose. Clearly, this depends on how much shorter than predicted by Minkowski's theorem we want the vectors to be. In our case, this depends on the concrete imbalance of the given parameters N_1 and N_2 in the range $N_1^3 > N_2 > N_1^{2.5}$. Finally, note that increasing e_0 comes at the cost of slightly decreasing the area of the rectangle containing our potential solutions. Thus, we expect the probability of finding lattices with suitable vectors to be slightly larger for smaller e_0.

We summarise our idea to solve Eq. (3.3) in Algorithm 3.1 for the case where $p = N_1 N_2$ and $N_1^{2.5} \approx N_2$, which we will analyse afterwards. Note that the algorithm generalises to more imbalanced parameters, too.

3.4.2 Analysis

We give a brief sketch for the analysis of the algorithm here.

On the limits of using random lattices

Algorithm 3.1 raises the question of how many lattices one expects to sample to obtain a lattice with a vector (e_1, d) such that $e_1 < B_1,\ d < B_2$ for fixed bounds B_1, B_2. Consider the problem where one samples many lattices of a given fixed determinant in $\mathbb{R}^2$ randomly and one looks for a particularly short vector in all the lattices sampled, where "short" is with respect to a weighted inner product. Since a weighted inner product can also be reduced to the usual euclidean norm by scaling the lattice appropriately, it suffices to consider the euclidean case. We use the following result from [AEN19, Sect. 4.1] for 2-dimensional lattices.

Lemma 3.4.1. *Let $Z_1, ...Z_k$ be the length of the shortest vectors in k independent random lattices of unit volume and $Z_{\min} := \min\{Z_1, ..., Z_k\}$, then $\mathbb{E}(Z_{\min}) \leq O\left(\frac{1}{\sqrt{k}}\right)$ for $k \geq 2$.*

Algorithm 3.1: Solving Eq. (3.3) using a modification of [dQKL+21, Alg. 1]

Input: (Imbalanced) SIDH parameters p, N_1, N_2 with $p = N_1N_2$ and $N_1^{2.5} \approx N_2$.
Output: A solution (a, b, c, d, e) to Eq. (3.3).

1 Set $e_0 := 2$.
2 **if** e_0p^{-1} *is a quadratic non-residue mod* N_1^2 **then**
3 Set $e_0 := e_0 + 1$ and go to Step 2.
4 Compute τ_0 such that $\tau_0^2 \equiv N_2^2 e_0 p^{-1} \bmod N_1^2$.
5 $\mathcal{L}_{e_0} := \mathrm{Span}_{\mathbb{Z}}((\sqrt{p}N_1^2, 0), (\sqrt{p}\tau_0, N_2\sqrt{e_0}))$.
6 Set $\mathbf{v} = (v_1, v_2)$ to be the shortest vector of $\mathcal{L}_{e_0}$ computed with Lagrange-Gauss.
7 **if** $\|\mathbf{v}\| \leq \sqrt{N_1}N_2$ **then**
8 Set $d := \frac{v_1}{\sqrt{p}}$, $e := \left(\frac{v_2}{N_2}\right)^2$.
9 **if** $eN_2^2 > pd^2$ **then**
10 $R := \frac{N_2^2 e - pd^2}{N_1^2}$.
11 Set $c := 1$.
12 **if** $R - pc^2 > 0$, $R - pc^2$ *is prime, and* $R - pc^2 \equiv 1 \bmod 4$ **then**
13 Find $(a, b) \in \mathbb{Z}$ such that $a^2 + b^2 = R - pc^2$.
14 **return** (a, b, c, d, e).
15 Set $c := c + 1$ and go to Step 12.
16 Set $e_0 := e_0 + 1$ and go to Step 2.

Looking at Algorithm 3.1, we see that the determinant of the lattices we consider will grow, since e_0 increases in the algorithm. Thus, we have to be a little bit more careful in the analysis. The analysis of the algorithm will rely on the heuristic that the set of lattices sampled scaled by their determinant behaves like a random unit lattice in $\mathbb{R}^2$.

Lemma 3.4.2. *Let k be the number of lattices sampled in Algorithm 3.1, $\mathcal{L}_{e_0}$ be the lattice defined in Step 5 of the algorithm and d_0 its determinant. Assume the shortest non-zero vectors of $\{\mathcal{L}_{e_0}/\sqrt{d_0}\}$ for e_0 as in Algorithm 3.1 behave like the one of a random unit lattice in $\mathbb{R}^2$ and let λ_1 denote the shortest non-zero vector of the lattices appearing in Algorithm 3.1. Then*

$$\lambda_1 \leq O^*\left(k^{-\frac{1}{4}}p^{\frac{1}{4}}N_1\sqrt{N_2}\right).$$

Proof. According to Lemma 3.4.1 and the assumption that the shortest non-zero vectors of $\{\mathcal{L}_{e_0}/\sqrt{d_0}\}$ behave like the ones of a random unit lattice in $\mathbb{R}^2$, we expect that there exists an e_0 such that

$$\lambda_1 \leq O^*\left(\sqrt{\frac{d_0}{k}}\right) = O^*\left(\frac{1}{\sqrt{k}}e_0^{\frac{1}{4}}p^{\frac{1}{4}}N_1\sqrt{N_2}\right) = O^*\left(k^{-\frac{1}{4}}p^{\frac{1}{4}}N_1\sqrt{N_2}\right),$$

where the last step used that e_0 is roughly a multiple of k determined by the proportion of quadratic residues in N_1^2 (e.g. this proportion is 1/8 if N_1 is a power of 2). □

Note that for the lattices considered in our algorithm, the second coordinate is always a multiple of $N_2\sqrt{e_0}$. This gives an increasing lower bound for the shortest non-zero vectors that could appear when sampling with increasing e_0. We can enforce, again using the fact that $e_0 \approx C \cdot k$ for a constant C, that

$$O^*(N_2\sqrt{e_0}) \leq O^*\left(k^{-\frac{1}{4}} p^{\frac{1}{4}} N_1 \sqrt{N_2}\right)$$

by bounding the number of lattices we sample in Algorithm 3.1 by

$$k \leq p^{1/3} N_1^{4/3} N_2^{-\frac{2}{3}}.$$

A rough estimate of the required precomputation

For the analysis of the algorithm, we will make use of the following heuristics.

1. For the first k lattices generated in Step 5 of Algorithm 3.1 with $k \leq p^{1/3} N_1^{4/3} N_2^{-\frac{2}{3}}$, the shortest vector of all these lattices follows the same distribution as for lattices of the same determinant sampled independently at random in $\mathbb{R}^2$.

2. The probability of $R - pc^2$ in Step 12 to be a prime and congruent to 1 mod 4 is expected to be the same as that of a random integer of the same size.

3. Half of the (e_0, e_1, d) with $d < \frac{N_2\sqrt{N_1}}{\sqrt{p}},\ e_1 < \frac{\sqrt{N_1}}{\sqrt{e_0}}$ satisfy $d < \frac{N_2\sqrt{e_0}e_1}{\sqrt{p}},\ e_1 < \frac{\sqrt{N_1}}{\sqrt{e_0}}$.

We discussed the first heuristic and why we impose the bound on k in the limitations of the random lattice model. Further, we verified experimentally that for the number of lattices sampled in testing our algorithm, no repetition of the lattices occurred. The second heuristic assumption has been used previously in the analysis of torsion point attacks [dQKL+21]. For the third heuristic, one can verify that the first inequalities define a rectangle of area $\frac{N_2 N_1}{\sqrt{pe_0}}$. The other inequalities can be rewritten as

$$e_1 < \frac{\sqrt{N_1}}{\sqrt{e_0}}, \quad e_1 N_2 \sqrt{e_0} - d\sqrt{p} > 0.$$

This defines the area of a triangle with vertices:

$$(0,0), \left(\frac{\sqrt{N_1}}{\sqrt{e_0}}, 0\right), \left(\frac{\sqrt{N_1}}{\sqrt{e_0}}, \frac{N_2\sqrt{N_1}}{\sqrt{p}}\right).$$

The area of the triangle is half of the one of the rectangle which is the rationale behind our heuristic. Note that in both cases we consider $\pm e_1$ and $\pm d$ as only their squares e_1^2 and d^2 appear in the norm equation we wish to solve.

Proposition 3.4.3. *Let $p = N_1 N_2$ with $N_1^{2.5} \approx N_2$. Under the heuristics stated at the beginning of this subsection, Algorithm 3.1 returns a solution (a, b, c, d, e) to Eq. (3.3) such that $e \leq N_1$ in time $O^*(\sqrt{N_1})$.*

Proof. It is a straightforward calculation and follows from our previous discussion that (a, b, c, d, e) as returned by Algorithm 3.1 is a solution to Eq. (3.3). In Step 7, since $v_2 \leq \|\mathbf{v}\| \leq \sqrt{N_1} N_2$, one has $e = \left(\frac{v_2}{N_2}\right)^2 \leq N_1$ as claimed.

We can efficiently compute square roots modulo N_1^2 and using the second heuristic, we expect Step 9 to succeed half the time. Further, for a fixed d, e, Step 13 of the algorithm is efficient as was shown by [dQKL+21], assuming $R - pc^2$ in Step 12 to be a prime congruent to 1 mod 4 with the same probability as a random integer of roughly the same size. Thus, we are left to show how many lattices we need to sample until we find a sufficiently short vector.

By Lemma 3.4.2, sampling k lattices below the bound given in the first heuristic, the expected shortest non-zero vector of all these lattices is of size approximately

$$O^*\left(k^{-\frac{1}{4}} p^{\frac{1}{4}} N_1 \sqrt{N_2}\right).$$

In order for this vector to be smaller than $N_2\sqrt{N_1}$, one needs to sample roughly $k \in O^*(pN_1^2 N_2^{-2}) = O^*(\sqrt{N_1})$ lattices, using $p = N_1 N_2$ and $N_1^{2.5} \approx N_2$. As this k is below the bound given by Heuristic 1, this finishes the proof. □

To execute the torsion point attacks, one would just apply the same strategy as described in the previous section due to [dQKL+21], recovering an e-isogeny using a meet-in-the-middle search to recover the sought secret isogeny.

Remark 3.4.4. For imbalanced parameters with $N_2 > N_1^3$, torsion point attacks were already known to be more efficient than plain meet-in-the-middle search due to [dQKL+21]. If the missing e-isogeny can be computed using meet-in-the-middle, then Proposition 3.4.3 shows how to extend these torsion attacks to $N_2 \approx N_1^{2.5}$. Note that for the range where $N_1^3 > N_2 > N_1^{2.5}$ the same algorithms outlined in this section still apply, however the closer N_2 is to N_1^3, the fewer lattices need to be sampled to obtain an e smaller than N_1. Sampling fewer lattices decreases the cost of the precomputation.

We have only discussed the size of e as an indicator for the hardness of recovering an e-isogeny. However, in practice one needs to be slightly more careful since the

computation of an isogeny of smooth degree is significantly more efficient. We leave a more thorough theoretical analysis to future work, but the following example shows that even for cryptographic sizes sufficiently smooth e can be generated.

3.4.3 Experiments

We implemented Algorithm 3.1 in MAGMA [BCP97] and we tested the method for imbalanced but not overstretched parameters, i.e. with $p \approx N_1 N_2$.

To show that the methods outlined in this section can be used to lower the required imbalance for torsion point attacks we conducted experiments on the following toy parameter set. We set $N_1 := 2^{108}$, $N_2 := 3^{195}$ and $p = 79 \cdot N_1 N_2 + 1$. Note that in this case $N_2 \approx N_1^{2.8}$. Among 2^{31} lattices generated for different e_0, 46 contained a vector satisfying the bounds of (3.7). The smoothest of the solutions to the norm equation had a 2^{14}-smooth e_0 and a 2^{10}-smooth e_1.

Taking parameters of cryptographic size, we chose $N_1 := 2^{216}$, $N_2 := 3^{400}$, and $p := 12 N_1 N_2 - 1$. Note that N_1 is exactly the same as in SIKE434 (N_2 and p are different to obtain the imbalance). Assuming that 72 bits of the secret are known due to guessing or due to leakage, we are left to recover an isogeny of degree 2^{144}. We generated lattices for 2^{34} choices of e_0, where we iterated through $7 + 8 \cdot i$ for $i \in \{0, \ldots, 2^{34}\}$. This choice ensures that e_0/p is a quadratic residue modulo 2^{144}.

Among the corresponding lattices, we found 245 of them to contain a vector (d, e_1) lying in the rectangle specified by the bounds of (3.7). Of those lattice vectors, 122, i.e. almost exactly half of them, corresponded to a solution to the norm equation as predicted by Heuristic 3. Considering all of these solutions, we get multiple reasonably smooth solutions to compute e_0- and e_1-isogenies. For example, we have a solution for $e_0 := 69397070847$, which is 2^{12}-smooth, with $e_1 := 240160$, which is 2^7-smooth. These values correspond to

$$\begin{aligned} d =& 10993831400842087678858215001111474291441990527789812293144468943848641056. \end{aligned}$$

For these parameters, the representation of

$$\frac{N_2^2 e_0 e_1^2 - p d^2}{N_1^2 2^{144}}$$

as $a^2 + b^2 + pc^2$ is given with a, b, c as

$$\begin{aligned} a =& 1328805005060809160671576896963371580297932031578472112138136767857609525428166156088609819903553426283421929680999848530334697699583470205057744184768664389 33, \\ b =& 2709830660694717233733080620065189566450824856002448010392947012291915646053894775991951846575047145368538510050311217377778393928141714324565726194192617494 8, \\ c =& 829. \end{aligned}$$

Isogenies of the given degree can still be computed efficiently using [BDLS20]. Note that the parameters used for this example are still very imbalanced with $N_2 \approx N_1^{2.93}$, yet slightly less imbalanced than predicted by previous work. Further, we were far from exhausting the search space for both of the examples given above and one would likely be able to find smaller and smoother e.

3.5 Quantum hidden shift attacks on SIDH

Unlike other isogeny-based protocols, SIDH was widely believed to be immune to subexponential quantum attacks because of the non-commutative structure of the endomorphism rings of supersingular curves. In particular, many believed that no reasonable variant of Childs–Jao–Soukharev's attack [CJS14] applies in the case of SIDH [JD11, p. 18, Sect. 5], since it crucially relies on the commutativity of the ideal class group action.

In this section, we contradict this commonly believed misconception for SIDH with overstretched and imbalanced parameters. We provide a new quantum attack on these SIDH variants which uses a reduction of the underlying computational problem to an injective abelian hidden shift problem. This can be solved in quantum subexponential time using variants of an algorithm by Kuperberg [Kup05] and thus disproves the common belief mentioned previously.

Let $\varphi : E_0 \to E_0/K$ be a secret isogeny that an attacker wishes to recover. As in SIDH, let E_0, E_0/K, $\deg(\varphi)$, and certain torsion point images under the secret isogeny be known publicly. The idea underlying our reduction is to construct an abelian group $G \subset \mathrm{End}(E_0)$ acting freely and transitively on certain cyclic subgroups of E_0. These subgroups are kernels of $\deg(\varphi)$-isogenies, and therefore they can be mapped to supersingular elliptic curves $\deg(\varphi)$-isogenous to E_0. The group action of G can then be understood as an

action on the curves themselves. Forcing the endomorphisms in G to be of a certain degree, the public torsion point information allows an adversary to compute the action on E_0/K efficiently under some heuristics. Finally, solving an abelian hidden shift problem of two functions mapping G to a set of curves $\deg(\varphi)$-isogenous to E_0 containing E_0/K enables an attacker to recover K and therefore φ. This is a different way of exploiting torsion point information compared to the attacks outlined in previous sections.

While this attack does not threaten SIDH with balanced parameter sets as originally proposed by Jao and De Feo [JD11], it shows that an attack using a hidden shift algorithm is possible despite SIDH's non-commutative nature.

We describe our new attack as a special instance in a more general setting which might be of independent interest beyond isogeny-based cryptography. Further, it allows us to unify this new reduction to a hidden shift problem with prior quantum attacks on isogeny-based schemes such as the one due to Childs, Jao and Soukharev which constructs isogenies between ordinary curves [CJS14], or similar applications of quantum hidden shift algorithms to CSIDH [BS20, Pei20].

In Section 3.5.1, we briefly review the complexity of the quantum algorithms used in our attack. In Section 3.5.2, we present our general framework, namely sufficient conditions for computing preimages of one-way functions via reduction to a hidden shift problem, and then present the general idea for our new attack on overstretched SIDH in Section 3.5.3. We then make some modification to this general approach. We describe explicitly the groups acting in Section 3.5.4 and how the group action can be computed by lifting endomorphisms in Section 3.5.5. We summarise the resulting algorithm in Section 3.5.6. Finally, we illustrate how our general framework can be instantiated with the attack of Childs, Jao and Soukharev and its generalisation to CSIDH in Section 3.5.7.

3.5.1 Quantum algorithms to solve hidden shift problems

First, we recall what is meant when two functions are said to be shifts of each other, or equivalently that these two functions *hide a shift*.

Definition 3.5.1. Let $F_0, F_1 \colon G \to X$ be two functions from a group G to a set X such that there exists some $s \in G$ satisfying $F_0(g) = F_1(g \cdot s)$ for all $g \in G$. The *hidden shift problem* is to find s given oracle access to the functions F_0 and F_1.

Multiple approaches utilizing quantum computation have been proposed to solve the hidden shift problem. Some of these works have considered different group structures as well as variations on the premise. We summarise some quantum algorithms solving the

injective abelian hidden shift problem, i.e., where the functions F_i are injective functions and G is abelian.

The first quantum subexponential algorithm is due to Kuperberg [Kup05] and reduces the hidden shift problem to the hidden subgroup problem in the dihedral group, i.e. to finding a subgroup of the dihedral group such that a function (obtained from combining the input functions of the hidden shift problem) is constant exactly on its cosets. It requires $2^{O(\sqrt{\log |G|})}$ quantum queries, for a finite abelian group G. A modification of this method proposed by Regev [Reg04] reduces the memory required by Kuperberg's approach (from super-polynomial to polynomial) while keeping the running time quantum subexponential. Another, slightly faster, algorithm is the collimation sieve proposed by Kuperberg, which uses polynomial quantum space [Kup11]. In this variant, parameter trade-offs between classical and quantum running time and quantum accessible memory are possible.

These algorithms to solve the hidden shift problem when G is abelian generally begin by producing some random quantum states, each with an associated classical "label", by evaluating the group action on a uniform superposition over the entire group G. For this generation of states, oracle access to the two functions F_0 and F_1 is needed. Then, the hidden shift s is extracted bitwise by performing measurements on specific quantum states (i.e., ones with desirable labels) which are generated from the random states via a sieve algorithm.

3.5.2 Malleability oracles and hidden shift attacks

In this section, we introduce the notion of a *malleability oracle* for a one-way function. Under some conditions, such an oracle allows us to compute preimages in quantum subexponential time via a reduction to the hidden shift problem.

First, we define an oracle capturing the main premise required for our strategy to compute preimages of one-way functions.

Definition 3.5.2. Let $f : \mathcal{I} \to \mathfrak{J}$ be an injective (one-way) function between sets and let $\cdot$ be the action of a group G on $\mathcal{I}$. A *malleability oracle* for G at $o := f(i)$ provides the value of $f(g \cdot i)$ for any input $g \in G$, i.e., the malleability oracle evaluates the map

$$g \mapsto f(g \cdot i).$$

We call the function f *malleable* if a malleability oracle is available at every $o \in f(\mathcal{I})$.

To abstract away from the notion of group actions, it would be possible to define malleability in terms of more general knowledge relating inputs and outputs of a one-way function. Yet, in the following we concentrate on group actions as defined above.

In Section 3.5.3, we show how a polynomial-time malleability oracle can be constructed in the context of SIDH with overstretched parameters, and in Section 3.5.7 we describe other schemes where such oracles arise naturally.

Reduction to the hidden shift problem

Given a malleability oracle at $o = f(i)$, computing a preimage of o reduces to a hidden shift problem in the following case.

Proposition 3.5.3. *Let $f : \mathcal{I} \to \mathfrak{J}$ be an injective (one-way) function between sets and let G be a group acting transitively on $\mathcal{I}$. Given a malleability oracle for G at $o := f(i)$, the preimage of o can be computed by solving a hidden shift problem.*

Proof. Let k be an arbitrary but fixed element in $\mathcal{I}$ and define

$$F_k \ : \ G \to \mathfrak{J} \ , \ \theta \mapsto f(\theta \cdot k).$$

Since f is an injective function, $i = f^{-1}(o)$ is unique and thus F_i is well-defined. Moreover, the malleability oracle allows us to evaluate the function F_i on any $\theta \in G$, as $F_i(\theta) = f(\theta \cdot i)$.

Fix some arbitrary $j \in \mathcal{I}$. Since we know j, we can evaluate F_j on any group element θ by evaluating $f(\theta \cdot j)$ via simply computing the group action. Due to the transitivity of the group action of G, there exists $\sigma \in G$ such that $i = \sigma \cdot j$. Since for all $\theta \in G$

$$F_i(\theta) = f(\theta \cdot i) = f(\theta\sigma \cdot j) = F_j(\theta\sigma),$$

the functions F_j and F_i are shifts of each other. Hence, solving the hidden shift problem for F_i and F_j allows us to recover σ, and thus to compute $i = \sigma \cdot j$. □

The following corollary will be used in our attack on overstretched SIDH.

Corollary 3.5.4. *Let $f : \mathcal{I} \to \mathfrak{J}$ be an injective (one-way) function between sets and let G be a finitely generated abelian group acting freely and transitively on $\mathcal{I}$. Given a malleability oracle for G at $o := f(i)$, the preimage of o can be computed in quantum subexponential time.*

Proof. Given Proposition 3.5.3 and the discussion in Section 3.5.1, we only need to show that for every $k \in \mathcal{I}$ the function $F_k(\theta) = f(\theta \cdot k)$ is injective. Then we are left with an

instance of the injective abelian hidden shift problem which can be solved in quantum subexponential time with an algorithm such as Kuperberg's.

Suppose that $F_k(g) = f(g \cdot k) = f(h \cdot k) = F_k(h)$ for some $g, h \in G$. Since f is injective and the group action is free, this implies $g = h$. □

3.5.3 Quantum subexponential time attack on overstretched SIDH

Despite the non-commutative nature of SIDH, we show in this section that one can find an abelian group action on its private key space. Moreover for sufficiently *overstretched* SIDH parameters, the torsion point information revealed in the protocol allows us to build a malleability oracle for this group action. This gives rise to an attack using quantum subexponential hidden shift algorithms as outlined in the previous subsection.

First, we give an overview of our approach to exploit the torsion point information. We then solve some technical issues which require small tweaks to this general strategy in the following two subsections.

Throughout this section, we use the following notation. Let $p \equiv 3 \pmod 4$ be prime, let E_0 be the supersingular elliptic curve with j-invariant 1728 defined over $\mathbb{F}_p$, given by the equation $y^2 = x^3 + x$, and let $\mathcal{O}_0 = \mathrm{End}(E_0)$ be its well known endomorphism ring (see Example 2.2.25).

Remark 3.5.5. The attack we describe can be expanded to other curves that are close to E_0 by computing the isogeny to E_0 and translating the problem to E_0. This applies for example to the curve

$$E_6 : y^2 = x^3 + 6x^2 + x,$$

the unique curve 2-isogenous to E_0 which is used as the starting curve in the updated parameters of SIKE for the second round of NIST's post-quantum standardisation effort [JAC+19].

Overview of the attack

Let $\mathcal{I}$ be the set of cyclic N_1-order subgroups of E_0, and let $\mathfrak{J}$ be the set of j-invariants of all supersingular curves that are N_1-isogenous to E_0. Let f be the function sending any element of $\mathcal{I}$ to the j-invariant of the codomain of its corresponding isogeny, i.e.,

$$f : \mathcal{I} \to \mathfrak{J}, \quad K \mapsto j(E_0/K). \tag{3.8}$$

The function f can be efficiently computed on any input using Vélu's formulae [Vél71], provided N_1 is sufficiently smooth and that the N_1-torsion is defined over a sufficiently small extension field of $\mathbb{F}_p$. In SIDH, the latter is achieved by choosing $N_1|p-1$, but this is true more generally for sufficiently powersmooth N_1.

On the other hand, inverting f amounts to finding an isogeny of degree N_1 from E_0 to a curve in a given isomorphism class, or equivalently to finding the subgroup of E_0 defining this isogeny, which is a variant of the pure isogeny problem with known degree.

As modelled by the SSI-T problem, the SIDH protocol provides additional torsion point information. We restate solving this problem as the following task in the notation we will use throughout this section.

Task 3.5.6. *Let p be a large prime, let N_1 and N_2 be two smooth coprime integers such that $E_0[N_1]$ and $E_0[N_2]$ can be represented efficiently, let $K \in \mathcal{I}$ be a cyclic subgroup of order N_1 of E_0 chosen uniformly at random, and let $\varphi : E_0 \to E_0/K$. Given the supersingular elliptic curves E_0 and E_0/K together with the restriction of φ to $E_0[N_2]$, compute K.*

Our attack will exploit the information provided by the restriction of the secret isogeny to $E_0[N_2]$ to construct a malleability oracle for f at the (unknown) secret. Following the framework outlined in Section 3.5.2, this gives rise to an attack on *overstretched* SIDH.

Let G be a subgroup of $(\mathcal{O}_0/N_1\mathcal{O}_0)^*$. Then G induces a group action on $\mathcal{I}$ given by

$$G \times \mathcal{I} \to \mathcal{I} \ , \ (\theta, K) \mapsto \theta(K).$$

Indeed, the degree of any non-trivial representative θ is coprime to N_1 and thus preserves the order of any generator of K.

Note that the full group $(\mathcal{O}_0/N_1\mathcal{O}_0)^*$ is not abelian. Our attack will require an abelian subgroup G acting on $\mathcal{I}$ such that G acts freely and transitively on the orbit of an isogeny kernel of an isogeny $E_0 \to E_0/K$ under this group action, as well as one element in this orbit. This leads to the following task.

Task 3.5.7. *Let $K \in \mathcal{I}$ be any cyclic subgroup of E_0 of order N_1 chosen uniformly at random and let $\varphi : E_0 \to E_A := E_0/K$. Compute an element $L \in \mathcal{I}$ and an abelian subgroup G of $(\mathcal{O}_0/N_1\mathcal{O}_0)^*$ such that G acts freely and transitively on the orbit $G \cdot L$, f as defined in (3.8) is injective on $G \cdot L$ and $j(E_A)$ is contained in $f(G \cdot L) \subset \mathfrak{J}$.*

We will solve this task by finding three subsets of $\mathcal{I}$ restricted to which f is injective, and we give abelian groups that induce the required action on these subsets. Furthermore, the image of f restricted to one of these three subsets of $\mathcal{I}$ will always contain $j(E_0/K)$.

In order to apply our general framework from Section 3.5.2, it remains to construct a malleability oracle for f at $j(E_0/K)$ for any secret $K \in \mathcal{I}$. To construct this oracle, we use both the torsion point information provided in the SIDH protocol and a solution to the following task.

Task 3.5.8. *Given an endomorphism $\theta \in G$ of degree coprime to N_1 and an integer N_2 coprime to N_1, compute an endomorphism θ' of degree N_2 such that θ and θ' induce the same action on the set $\mathcal{I}$ of cyclic subgroups of $E_0[N_1]$ of order N_1.*

In [KMPW21, Appx. C], we give a direct solution to a variation of this task when using sufficiently overstretched and imbalanced parameters, i.e. $N_2 > p^2 N_1^4$. However, we will show that using the Frobenius map it suffices to lift elements of πG, where π is the Frobenius map. We describe a solution to Task 3.5.8 for these elements requiring only $N_2 > pN_1^4$ to lift endomorphisms from $\pi\mathbb{Z}[\iota]$ to an element of norm eN_2 in Section 3.5.5.

Due to the coprimality of $\deg(\theta)$ and N_1, the following lemma, depicted in Fig. 3.2, follows (see also Fig. 2.2).

Lemma 3.5.9. *Let $\varphi : E_0 \to E_A$ be an isogeny of degree N_1 and let $\theta \in \mathrm{End}(E_0)$ be of degree coprime to N_1. Then $E_A/\varphi(\ker\theta)$ is isomorphic to $E_0/\theta(\ker\varphi)$.*

$$\begin{array}{ccc} E_0 & \xrightarrow{\quad\varphi\quad} & E_A \\ \Big\downarrow{\scriptstyle\theta} & & \Big\downarrow{\scriptstyle[\varphi]_*\theta} \\ E_0 & \xrightarrow[[\theta]_*\varphi]{} & E_0/\theta(\ker\varphi) \cong E_A/\varphi(\ker\theta) \end{array}$$

Fig. 3.2 The isogeny φ and the endomorphism θ have coprime degrees and thus the diagram commutes.

Let N_3 be the degree of θ. We cannot compute the curve $E_0/\theta(\ker\varphi)$ in general without the knowledge of the isogeny φ or its action on the N_3-torsion. However, we can compute the curve if we find an endomorphism θ' of degree N_3' such that θ and θ' have the same action on the N_1-torsion and $\varphi|_{E_0[N_3']}$ is known. This is the motivation behind Task 3.5.8, as we know the action of φ on the N_2-torsion. A solution to this task yields a malleability oracle for f with respect to the previously described group action of G on $\mathcal{I}$ in the SIDH setting.

We outline the construction of the malleability oracle in Algorithm 3.2. Correctness will follow from the proof of Proposition 3.5.30 given a suitable choice of the acting group G which we will discuss in Section 3.5.4.

Algorithm 3.2: Computation of $f(\theta(K))$, given $f(K)$ and $\theta \in G$

Let $\varphi : E_0 \to E_A := E_0/K$ be an isogeny of degree N_1, let N_2 be coprime to N_1 and $G \subset (\mathcal{O}_0/N_1\mathcal{O}_0)^*$ one of the abelian groups as in Task 3.5.7 that acts freely and transitively on K.

Input: E_0, $f(K) = j(E_A)$, $\varphi|_{E_0[N_2]}$ and $\theta \in G$.

Output: $f(\theta(K)) = j(E_0/\theta(K))$.

1 Compute endomorphism θ' of degree N_2 having the same action as θ on cyclic N_1-order subgroups of $E_0[N_1]$ as provided by a solution to Task 3.5.8.

2 Determine $\varphi(\ker\theta')$, using the knowledge of φ on $E_0[N_2]$.

3 Compute $j(E_A/\varphi(\ker\theta')) = j(E_0/\theta(K))$.

4 **return** $f(\theta(K)) = j(E_0/\theta(K))$.

For parameters that allow us to construct a malleability oracle, we can then solve Task 3.5.6 underlying SIDH-like protocols via a reduction to an injective abelian hidden shift problem using the general reduction outlined in Section 3.5.2.

Informal result 3.5.10. *Suppose the parameters allow the efficient solution of Task 3.5.8, then Task 3.5.6 can be solved in quantum subexponential time.*

We use the remainder of this section to prove this result formally under certain assumptions. To this end, we first give a solution to Task 3.5.7. Then, for some parameters, we provide a solution to a variant of Task 3.5.8. More precisely, we will show using the Frobenius map that instead of lifting elements from G we can lift elements from πG. For this case, we then give a lifting procedure requiring overstretched and imbalanced parameters. We construct a malleability oracle using the torsion point information provided in SIDH and the subroutine solving our variant of Task 3.5.8. Apart from some technical details that we will address in the following, the informal result follows from Corollary 3.5.4.

Algorithm 3.3 gives an overview over the attack and Fig. 3.3 depicts the intuition behind it.

3.5.4 An effective free and transitive group action

Recall that E_0 is a supersingular curve with j-invariant 1728. In this section, we provide a solution to Task 3.5.7. For simplicity, we treat N_1 as a power of 2, but the results generalise to any power of a small prime. A generalisation to powers of 3 is sketched in [KMPW21, Appx. B].

We provide the solution by identifying three subsets of $\mathcal{I}$ that are orbits under a free and transitive action of abelian subgroups of $(\mathcal{O}_0/N_1\mathcal{O}_0)^*$. More precisely, let $P \in E_0$

Algorithm 3.3: Solving SIDH's underlying hardness assumption via an abelian hidden shift problem

Let $\varphi : E_0 \to E_0/K$ be an N_1-isogeny and $N_2 \in \mathbb{Z}$ such that $\gcd(N_1, N_2) = 1$.
Input: E_0, E_0/K, $\varphi(E_0[N_2])$.
Output: Isogeny $E_0 \to E$, where $j(E) = j(E_0/K)$.

1 Compute an abelian group $G \subset (\mathcal{O}_0/N_1\mathcal{O}_0)^*$ acting freely and transitively on the orbit $G(K)$ and some $J \in G(K) \subset \mathcal{I}$.
2 Define $F_K : G \to \mathfrak{J}, g \mapsto f(g(K))$ and $F_J : G \to \mathfrak{J}, g \mapsto f(g(J))$.
3 Compute injective abelian hidden shift $\theta \in G$ of F_K and F_J, i.e., $\theta \in G$ such that $F_K(g) = F_J(\theta g)$ for all $g \in G$, using a quantum algorithm such as Kuperberg's. To this end, evaluate F_K using Algorithm 3.2 and F_J using the knowledge of J.
4 **return** Isogeny $E_0 \to E_0/\theta(J)$.

such that $\langle P, \iota(P)\rangle = E_0[N_1]$, where ι denotes the automorphism $(x, y) \mapsto (-x, iy)$ of E_0. Let $Q := P + \iota(P)$ and define the following three subsets of $\mathcal{I}$.

$$\mathcal{I}_1 := \{\langle P + [\alpha]\iota(P)\rangle \mid \alpha \text{ even}\}$$
$$\mathcal{I}_2 := \left\{\langle Q + \alpha\iota(Q)\rangle \mid \alpha \text{ even and } \alpha \in \left[0, \frac{N_1}{2} - 1\right]\right\}$$
$$\mathcal{I}_3 := \left\{\langle Q + \alpha\iota(Q)\rangle \mid \alpha \text{ even and } \alpha \in \left[\frac{N_1}{2}, N_1 - 1\right]\right\}$$

Recall the function f defined in (3.8), mapping cyclic subgroups of $E_0[N_1]$ of order N_1 to j-invariants of curves at distance N_1 from E_0,

$$f : \mathcal{I} \to \mathfrak{J}, \quad K \mapsto j(E_0/K).$$

We will show that restricting the function f to any of the subsets $\mathcal{I}_1$, $\mathcal{I}_2$, or $\mathcal{I}_3$ yields an injective function and we will prove that $f(\cup_i \mathcal{I}_i) = f(\mathcal{I})$. Furthermore, we will see that

$$G_0 := \{a + b\iota \in (\mathbb{Z}[\iota]/N_1\mathbb{Z}[\iota])^* \mid a \text{ odd}, b \text{ even}\}$$

acts transitively on $\mathcal{I}_1$. In order to ensure that the action is free, we identify two endomorphisms $a + b\iota$ and $a' + b'\iota$ in G_0 if there exists an odd $\lambda \in \mathbb{Z}/N_1\mathbb{Z}$ such that $a \equiv \lambda a' \pmod{N_1}$ and $b \equiv \lambda b' \pmod{N_1}$. We denote the resulting group by G.

In order to obtain free and transitive group actions on $\mathcal{I}_2$, and $\mathcal{I}_3$, we define similarly

$$H_0 := \{a + b\iota \ \in (\mathbb{Z}[\iota]/(N_1/2)\mathbb{Z}[\iota])^* \mid a \text{ odd}, b \text{ even}\}.$$

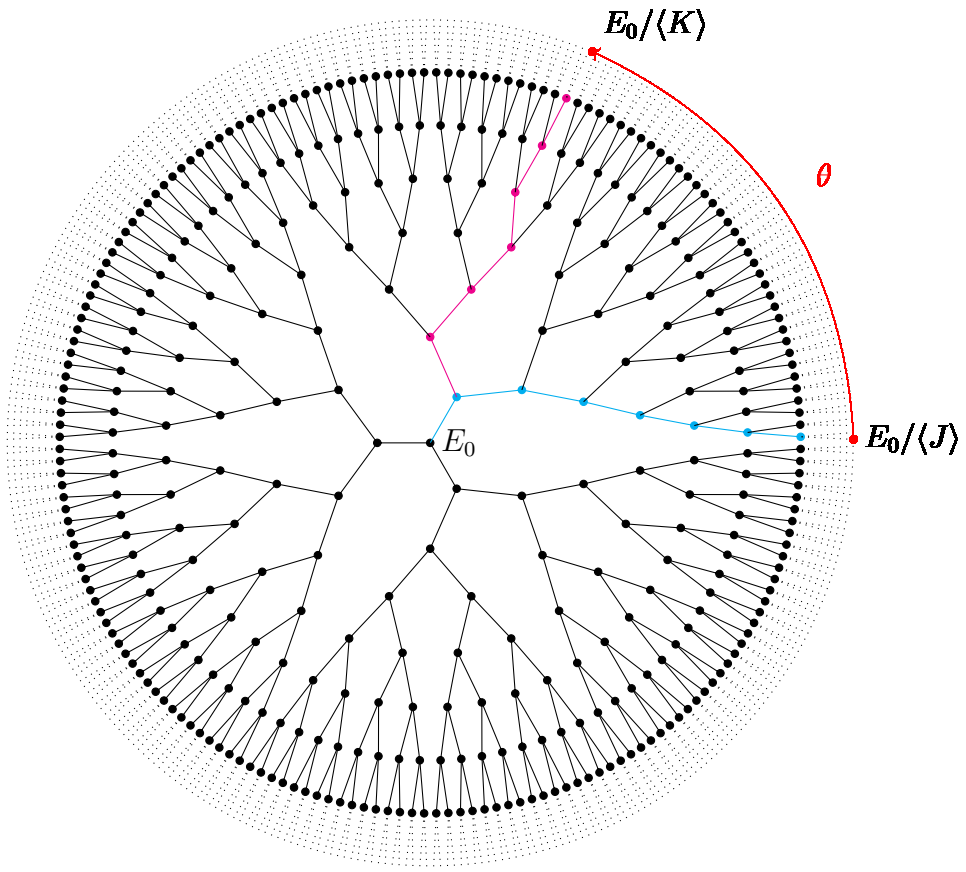

Fig. 3.3 Intuition behind the hidden shift attack: The group induces an action on the curves at distance N_1 from E_0. Finding the hidden shift θ allows us to *shift* a known isogeny $E_0 \to E_0/\langle J\rangle$ to the secret path $E_0 \to E_0/\langle K\rangle$ and thus to recover K.

Again, we identify two endomorphisms $a + b\iota$ and $a' + b'\iota$ in H_0 if there exists an odd $\lambda \in \mathbb{Z}/(N_1/2)\mathbb{Z}$ such that $a \equiv \lambda a' \pmod{N_1/2}$ and $b \equiv \lambda b' \pmod{N_1/2}$, we obtain a group H. The group H will act freely and transitively on $\mathcal{I}_2$ and $\mathcal{I}_3$.

In summary, one of these three options will always be a solution to Task 3.5.7.

The map f is based on the well-known correspondence between $\mathcal{I}$ and curves at distance N_1 from E_0. However, this correspondence is not necessarily one-to-one. In particular, if E_0 has a non-scalar endomorphism of degree N_1^2, then that endomorphism can be decomposed as $\hat{\tau}_1 \circ \tau_2$, where τ_1 and τ_2 are non-isomorphic isogenies of degree N_1 from E_0 to the same codomain E. However, for small enough N_1 the following lemma shows that two kernels correspond to the same curve if and only if they are linked by the automorphism ι.

Lemma 3.5.11. *Suppose that $N_1^2 < \frac{p+1}{4}$. Then the only endomorphisms of degree N_1^2 of E_0 are $[N_1]$ and $[N_1] \cdot \iota$, where $\iota : E_0 \to E_0, (x, y) \mapsto (-x, iy)$ is the non-trivial automorphism on E_0.*

Proof. Due to the condition $N_1^2 < \frac{p+1}{4}$, an endomorphism θ of degree N_1^2 lies in $\mathbb{Z}[\iota]$. Let $\theta = a + b\iota$ for some $a, b \in \mathbb{Z}$. Then the degree of θ is $a^2 + b^2$. We are left to prove that the only ways to decompose N_1^2 as a sum of two squares are trivial, i.e.,

$$N_1^2 = N_1^2 + 0^2 = 0^2 + N_1^2.$$

Let $N_1 = 2^k$. We prove the statement by induction on k. For $k = 1$ the statement is trivial. Suppose that $k > 1$ and that $N_1^2 = a^2 + b^2$. Then a and b cannot both be odd as N_1^2 is divisible by four. If they were both even, then dividing by four yields a decomposition of $(N_1/2)^2 = (a/2)^2 + (b/2)^2$. By the induction hypothesis, this decomposition is trivial implying that N_1^2 can also only be decomposed in a trivial way. □

Corollary 3.5.12. *Suppose that $N_1^2 < \frac{p+1}{4}$. Let ϕ and ϕ' be two isogenies of degree N_1 from E_0 to a curve E. Then either* $\ker \phi = \ker \phi'$ *or* $\ker \phi = \iota(\ker \phi')$.

Proof. Consider the endomorphism $\tau = \hat{\phi}' \circ \phi$ of E_0. The degree of τ is N_1^2, so $\tau = [N_1]$ or $\tau = [N_1] \cdot \iota$ by Lemma 3.5.11. In the former case, the isogenies ϕ and ϕ' are identical by the uniqueness of the dual. In the latter case, we have $\ker \phi = \iota(\ker \phi')$. □

Thus, an element in the image of f has precisely one preimage if the kernel of the corresponding isogeny is fixed by the automorphism ι.

Identifying an abelian group with $\mathcal{I}_1$

Now, we will give the free and transitive group action on $\mathcal{I}_1$ and show that f restricted to $\mathcal{I}_1$ is injective. Let P be a point such that $\langle P, \iota(P) \rangle = E_0[N_1]$ and recall

$$\mathcal{I}_1 := \{\langle P + [\alpha]\iota(P) \rangle \mid \alpha \text{ even}\}.$$

We show that the restriction of f to $\mathcal{I}_1$ is injective.

Proposition 3.5.13. *Let $j(E_0) = 1728$ and suppose that $N_1^2 < \frac{p+1}{4}$. The restriction of f to $\mathcal{I}_1$ is injective.*

Proof. We apply Corollary 3.5.12 to show that the codomains of isogenies with kernel in $\mathcal{I}_1$ are pairwise non-isomorphic curves. It is clear that $P + \alpha\iota(P)$ and $P + \alpha'\iota(P)$ are not scalar multiples of each other if $\alpha \neq \alpha'$ as $P, \iota(P)$ generate $E_0[N_1]$. It remains to show

that for any even α, α', the points $P + \alpha\iota(P)$ and $-\alpha' P + \iota(P)$ are not scalar multiples of each other. Note that we can restrict to odd λ as the order of both points is N_1 which we assumed to be a power of 2 in this section. Suppose there exists an odd λ such that

$$P + \alpha\iota(P) = \lambda(-\alpha' P + \iota(P)).$$

Since $\{P, \iota(P)\}$ is a basis of the N_1-torsion, this implies that $1 \equiv -\lambda\alpha' \pmod{N_1}$. Since α' is even this is a contradiction, concluding the proof. □

Clearly, $f(\mathcal{I}_1)$ does not include all elliptic curves at distance N_1 from E_0, i.e., all curves in $f(\mathcal{I})$. Every curve at distance N_1 from E_0 is of the form $E_0/\langle P + \alpha\iota(P)\rangle$ for some $\alpha \in \mathbb{Z}/N_1\mathbb{Z}$, which follows from the observation that the curves $E_0/\langle \beta_1 P + \beta_2\iota(P)\rangle$ and $E_0/\langle -\beta_2 P + \beta_1\iota(P)\rangle$ are isomorphic since their kernels are linked by ι. We first restrict ourselves to define a free and transitive group action on $\mathcal{I}_1$ and define the free and transitive group action on the kernels corresponding to the remaining curves later.

Recall that E_0 is a curve with a well-known endomorphism ring (Example 2.2.25), and we are interested in the endomorphisms that are of degree coprime to N_1. While there are infinitely many such endomorphisms, we are only concerned with their action on $E_0[N_1]$, i.e., we are looking at the group $(\mathcal{O}_0/N_1\mathcal{O}_0)^*$ which is isomorphic to $\mathrm{GL}_2(\mathbb{Z}/N_1\mathbb{Z})$ [Voi21, p. 676]. Furthermore, we are only concerned with the action of the endomorphisms on $\mathcal{I}$, i.e., on cyclic subgroups of $E_0[N_1]$ of order N_1, and we can therefore identify even more endomorphisms with each other by the following lemma.

Lemma 3.5.14. *Let (a, b, c, d) and (a', b', c', d') be the coefficients of θ and θ' with respect to some $\mathbb{Z}$-basis of the endomorphism ring $\mathcal{O}_0$ of E_0, and let $\mathcal{I}$ be the set of cyclic N_1-order subgroups of $E_0[N_1]$. Then $\theta(K) = \theta'(K)$ for every $K \in \mathcal{I}$ if and only if there exists some $\lambda \in (\mathbb{Z}/N_1\mathbb{Z})^*$ such that*

$$(a, b, c, d) \equiv \lambda(a', b', c', d') \pmod{N_1}.$$

Proof. Considering the respective restrictions to $E_0[N_1]$, two endomorphisms are equal if they lie in the same class in $(\mathcal{O}_0/N_1\mathcal{O}_0)^*$. Moreover, let θ_1, θ_2 be two endomorphisms such that $\theta_1 = [\lambda]\theta_2$ for some integer λ, and let P be an element of order N_1. Since scalar multiplication commutes with any endomorphism, it is easy to see that $\theta_1(P)$ and $\theta_2(P)$ generate the same subgroup in $E_0[N_1]$ if and only if λ is coprime to N_1. □

Now, we are ready to give a solution to Task 3.5.7 if $K \in \mathcal{I}_1$.

Proposition 3.5.15. *Let G be the group of equivalence classes of elements*

$$\{a + b\iota \mid a \text{ odd}, b \text{ even}\} \subset (\mathbb{Z}[\iota]/N_1\mathbb{Z}[\iota])^* \subset (\mathcal{O}_0/N_1\mathcal{O}_0)^*,$$

where we identify two elements if and only if they differ by multiplication by an odd scalar modulo N_1. G is an abelian group and its action on $\mathcal{I}_1$ is free and transitive.

Proof. It is easy to see that the endomorphisms in $\mathbb{Z}[\iota]$ of degree coprime to N_1 form an abelian subgroup of $\mathcal{O}_0$. Using any basis for $E_0[N_1]$ of the form $\{P, \iota(P)\}$, we can write the elements of this subgroup as matrices of the form $\begin{pmatrix} a & b \\ -b & a \end{pmatrix}$, where a is odd and b is even. By identifying two endomorphisms $a_1 + b_1\iota$ and $a_2 + b_2\iota$ if there exists an integer λ coprime to N_1 and an endomorphism δ such that $a_1 - \lambda a_2 + (b_1 - \lambda b_2)\iota = N_1\delta$, which is possible by Lemma 3.5.14, we obtain G. As G is closed under multiplication and reduction modulo N_1, it is a subgroup of an abelian group and therefore abelian itself. Note that G contains all equivalence classes under Lemma 3.5.14 of endomorphisms of the form $a + b\iota$ for even b, independently of the chosen basis.

To examine the orbit of an element in $\mathcal{I}$, which is a cyclic subgroup of order N_1 of $E_0[N_1]$, under the action of G, it is sufficient to look at the orbit of a generator of this cyclic group in $\mathcal{I}$. We consider the orbit of P which has coordinates $(1, 0)$ with respect to our basis under the group action of G. The image of $(1, 0)$ under an element $\begin{pmatrix} 1 & b \\ -b & 1 \end{pmatrix}$ is $(1, b)$ and thus inspecting the cyclic subgroups of E_0 generated by these points, we get $G \cdot \langle P \rangle = \mathcal{I}_1$. □

Free and transitive group action on $\mathcal{I}_2$ and $\mathcal{I}_3$

So far we have defined a free and transitive group action on $\mathcal{I}_1$ and thus for the curves in $f(\mathcal{I}_1)$. However, when the secret kernel is generated by $P + \alpha\iota(P)$ with α odd, the curve $E_0/\langle P + \alpha\iota(P)\rangle$ is not contained in $f(\mathcal{I}_1)$. This is the case we handle next.

One can show that the action of the previously defined group G acting on curves at distance N_1 from E_0 considered via f has three orbits. We have already seen that $f(\mathcal{I}_1)$ is one orbit, but the cases with odd α will split into two orbits. Clearly, G cannot be free and transitive on both orbits, since the size of the orbits is smaller than the cardinality of the group. We avoid this issue by choosing a different (but related) group of cardinality $N_1/4$, acting on the curves corresponding to an odd α.

Lemma 3.5.16. *Let P be a point such that $\langle P, \iota(P)\rangle = E_0[N_1]$ and let $Q := P + \iota(P)$. Define*

$$\mathcal{I}_2 := \left\{ \langle Q + \alpha\iota(Q)\rangle \mid \alpha \text{ even and } \alpha \in \left[0, \frac{N_1}{2} - 1\right] \right\}$$
$$\mathcal{I}_3 := \left\{ \langle Q + \alpha\iota(Q)\rangle \mid \alpha \text{ even and } \alpha \in \left[\frac{N_1}{2}, N_1 - 1\right] \right\}.$$

The restrictions $f_{|\mathcal{I}_2}$ and $f_{|\mathcal{I}_3}$ of f to $\mathcal{I}_2$ and $\mathcal{I}_3$ are injective.

Proof. We show that two distinct isogenies with kernel both in $\mathcal{I}_2$ (or both in $\mathcal{I}_3$) map to two non-isomorphic curves. Let α, α' be such that $\langle Q + \alpha\iota(Q)\rangle$ and $\langle Q + \alpha'\iota(Q)\rangle$ are both in $\mathcal{I}_2$, or $\mathcal{I}_3$, respectively. Suppose there exists an odd λ such that

$$Q + \alpha\iota(Q) = \lambda(Q + \alpha'\iota(Q)).$$

This means $1 - \lambda \equiv 0 \pmod{N_1/2}$ and $\alpha - \lambda\alpha' \equiv 0 \pmod{N_1/2}$ which implies $\alpha \equiv \alpha' \pmod{N_1/2}$. We are left to show that $Q + \alpha\iota(Q)$ is never an odd multiple of $-\alpha Q + \iota(Q)$. Suppose there exists an odd λ such that

$$Q + \alpha\iota(Q) = \lambda(-\alpha' Q + \iota(Q)).$$

This implies $1 + \alpha'\lambda \equiv \alpha - \lambda \equiv 0 \pmod{N_1/2}$, which is a contradiction, since $\alpha - \lambda \equiv 0 \pmod{N_1/2}$ implies that λ is even while $1 + \alpha'\lambda \equiv 0 \pmod{N_1/2}$ implies that λ is odd. Therefore, the curves $E_0/\langle Q + \alpha\iota(Q)\rangle$ and $E_0/\langle Q + \alpha'\iota(Q)\rangle$ are pairwise non-isomorphic. □

Finally, we give a free and transitive group action on $\mathcal{I}_2$ and $\mathcal{I}_3$. We start by defining the acting group.

We identify two endomorphisms $a + b\iota$ and $a' + b'\iota$ if there exists an odd $\lambda \in \mathbb{Z}/(N_1/2)\mathbb{Z}$ such that $a \equiv \lambda a' \pmod{N_1/2}$ and $b \equiv \lambda b' \pmod{N_1/2}$ and we call the resulting group H_0. Let H be the subgroup of H_0 containing elements with even b.

Proposition 3.5.17. *H acts freely and transitively on $\mathcal{I}_2$ and $\mathcal{I}_3$.*

Proof. It is enough to show that H acts transitively on $\mathcal{I}_2$ and $\mathcal{I}_3$ because H, $\mathcal{I}_2$ and $\mathcal{I}_3$ have the same cardinality. From $(1 + \alpha\iota)Q = Q + \alpha\iota(Q)$ it follows that the orbit $H \cdot \langle Q\rangle$ contains every element in $\mathcal{I}_2$. Similarly, H acts transitively on $\mathcal{I}_3$ as

$$(1 + \alpha\iota)(Q + N_1\iota(Q)/2) = (1 - \alpha N_1/2)Q + (\alpha + N_1/2)\iota(Q) = Q + (\alpha + N_1/2)\iota(Q),$$

where $(\alpha N_1/2)Q = 0$ as α is even. □

The only thing left to show is that every curve $E_0/\langle P + \alpha\iota(P)\rangle$ with odd α has a j-invariant contained in $f(\mathcal{I}_2)$ or $f(\mathcal{I}_3)$.

Proposition 3.5.18. *Let α be an odd integer. Then $f(\langle P + \alpha\iota(P)\rangle)$ is contained in $f(\mathcal{I}_2)$ or $f(\mathcal{I}_3)$.*

Proof. Observe that

$$P + \alpha\iota(P) = \frac{1+\alpha}{2}(P + \iota(P)) + \frac{\alpha-1}{2}(-P + \iota(P)) = \frac{1+\alpha}{2}Q + \frac{\alpha-1}{2}\iota(Q).$$

The sum of $\frac{1+\alpha}{2}$ and $\frac{\alpha-1}{2}$ is odd and therefore one of the fractions is even while the other one is odd. If $\frac{\alpha-1}{2}$ is even, then it is clear that the curve is contained in $f(\mathcal{I}_2)$ or $f(\mathcal{I}_3)$. In the case where $\frac{1+\alpha}{2}$ is even, $E_0/\langle\frac{1+\alpha}{2}Q + \frac{\alpha-1}{2}\iota(Q)\rangle$ is isomorphic to $E_0/\langle\frac{1-\alpha}{2}Q + \frac{\alpha+1}{2}\iota(Q)\rangle$, as their kernels are related by ι, and thus the curve is contained in $f(\mathcal{I}_2)$ or $f(\mathcal{I}_3)$. □

In this subsection, we have identified three subsets of $\mathcal{I}$, restricted to which f is injective. Moreover, we have seen that the union $\cup_{i=1}^{3} f(\mathcal{I}_i)$ contains the j-invariants of all curves at distance N_1 from E_0. Finally, we gave an abelian subgroup of $(\mathcal{O}_0/N_1\mathcal{O}_0)^*$ for each of these subsets of $\mathcal{I}$ that acts freely and transitively on it. Thus, we solve Task 3.5.7 as long as one determines or guesses which of the three $f(\mathcal{I}_i)$ contains $j(E_0/K)$.

Using the Frobenius map

We described how to choose suitable abelian subgroups of $(\mathcal{O}_0/N_1\mathcal{O}_0)^*$ in order to solve Task 3.5.7 after guessing whether $j(E_0/K)$ is a j-invariant in $f(\mathcal{I}_1)$, $f(\mathcal{I}_2)$, or $f(\mathcal{I}_3)$.

The elements of the acting groups chosen as described in the previous section can be trivially lifted to $\mathbb{Z}[\iota] := \mathbb{Q}[\iota] \cap \mathcal{O}_0$. In [KMPW21, Appx. C], we showed how these representatives can be lifted directly to elements of norm N_2 or eN_2, where e is a small positive integer, whenever the SIDH parameters N_1 and N_2 are sufficiently overstretched and imbalanced with $N_2 > p^2N_1^4$. For these parameters, this solves a variation of Task 3.5.8.

In this section we reduce the required imbalance partially by proving that we can lift elements from $\pi\mathbb{Z}[\iota]$ instead. Assuming that $N_2 > pN_1^4$, we will show how endomorphisms from $\pi\mathbb{Z}[\iota]$ can be lifted efficiently to another endomorphism of norm N_2 or eN_2, for some small integer e, inducing the same action on $\mathcal{I}$ in Algorithm 3.4. Note that it is not possible to choose a group generated by an element in $\pi\mathbb{Z}[\iota]$ to solve Task 3.5.7 directly, acting freely and transitively on a large number of N_1-isogeny kernels, as any such element has multiplicative order at most 4.

As before, let $\varphi : E_0 \to E_0/K$ denote the secret N_1-isogeny we want to compute. Recall that to run our attack we need to be able to compute $E_0/\theta(K)$ for every θ in the groups G acting on $\mathcal{I}_1$, and H acting on $\mathcal{I}_2$ and $\mathcal{I}_2$. We have seen that we can represent θ as an element in $\mathbb{Z}[\iota]$.

Let π denote the Frobenius map. Assuming that we can lift $\pi\theta$ to an endomorphism of degree N_2 inducing the same action on $\mathcal{I}$, we can compute $E_0/\pi\theta(K)$ using knowledge of $\varphi(E_0[N_2])$ as described in the overview of our attack. Now let $B := \theta(K)$. Given $E_0/\pi(B)$, we can compute E_0/B using the Frobenius map as follows.

Lemma 3.5.19. *Let E be an elliptic curve defined over $\mathbb{F}_p$, π the Frobenius map and let $B \subset E$ be a cyclic subgroup. The curve $E/\pi(B)$ is isomorphic to the image of the Frobenius map on E/B.*

Proof. Let ϕ_1 be the isogeny with kernel B and ϕ_2 the isogeny with kernel $\pi(B)$. The isogeny ϕ_1 is separable and its kernel is contained in the kernel of $\phi_2 \circ \pi$. Then, there exists a unique isogeny $\psi : E/B \to E/\pi(B)$ satisfying $\phi_2 \circ \pi = \psi \circ \phi_1$ (see [Sil09, Cor. III. 4.11]), i.e., the following diagram commutes.

$$\begin{array}{ccc} E & \xrightarrow{\phi_1} & E/B \\ {\scriptstyle\pi}\downarrow & & \downarrow{\scriptstyle\psi} \\ E & \xrightarrow[\phi_2]{} & E/\pi(B) \end{array}$$

The degree of a composition of isogenies is the product of its factors which implies $\deg(\psi) = p$. Furthermore, ψ is not separable as the Frobenius map is not. As ψ can be decomposed as a composition of the Frobenius map and a separable isogeny (see [Sil09, Cor. II.2.12]), $\deg(\psi) = p$ implies that ψ must be a composition of Frobenius and an automorphism. Hence, E_0/B and $E_0/\pi(B)$ are connected by Frobenius. □

Lemma 3.5.19 implies that we can compute $E_0/\theta(K)$ by first computing $E_0/\pi\theta(K)$ and then applying the Frobenius map. This gives rise to the following strategy when constructing the malleability oracle.

Assume we want to compute $E_0/\theta(K)$ for some $\theta \in \mathbb{Z}[\iota]$ and unknown K, given the image of the N_2-torsion of the isogeny $\varphi : E_0 \to E_0/K$. Using the lifting algorithm we will describe in the following, we compute an endomorphism θ' of degree N_2 or eN_2 for a small e that induces the same action on $\mathcal{I}$ as $\pi\theta$. As described previously, the

torsion point information allows us to compute $E_0/\theta'(K) = E_0/\pi\theta(K)$. By Lemma 3.5.19, applying the Frobenius map yields $E_0/\pi\theta'(K) = E_0/\theta(K)$.

3.5.5 Lifting $\theta \in \pi\mathbb{Z}[\iota]$ to an endomorphism of norm eN_2

In this subsection, we present an efficient algorithm to lift an endomorphism from $\pi\mathbb{Z}[\iota] = \pi(\mathbb{Q}[\iota] \cap \mathrm{End}(E_0))$ to another endomorphism in $\mathrm{End}(E_0)$ of degree N_2 or eN_2 that induces the same action on $\mathcal{I}$, whenever $N_2 > pN_1^4$. Here, e is the smallest positive integer such that $eN_2/p(c_0^2 + d_0^2)$ is a quadratic residue modulo $2N_1$, where $\pi(c_0 + d_0\iota) \in \pi\mathbb{Z}[\iota]$ is the endomorphism we want to lift.

This will solve the following task, which is a variant of Task 3.5.8, efficiently.

Task 3.5.20. *Let N_1, N_2 be coprime integers such that $N_2 > pN_1^4$, let $\theta := \pi(c_0 + d_0\iota) \in \pi\mathbb{Z}[\iota]$ be an E_0-endomorphism of degree coprime to N_1 and let e denote the smallest positive integer such that $eN_2/p\,(c_0^2 + d_0^2) \pmod{2N_1}$ is a quadratic residue. Compute an endomorphism θ' of degree N_2 or eN_2 such that $\theta(K) = \theta'(K)$ for all $K \in \mathcal{I}$.*

We have discussed how to use Frobenius in order to lift $\pi(c_0 + d_0\iota)$ instead of $c_0 + d_0\iota$. Therefore, this task solves Task 3.5.8 up to the following two relaxations. First, we require N_2 to be sufficiently large and imbalanced compared to N_1. Second, we allow θ' to be either of degree N_2 or eN_2 for some small positive integer e.

We now describe an algorithm to solve Task 3.5.20. By Lemma 3.5.14, it suffices to solve the following task, which is similar to the problem solved at the core of the KLPT algorithm [KLPT14].

Task 3.5.21. *Given $\theta = a_0 + b_0\iota + (c_0 + d_0\iota)\pi$, find $\theta' = a_1 + b_1\iota + (c_1 + d_1\iota)\pi$ of degree N_2 or eN_2 with coefficients $(a_1, b_1, c_1, d_1) \equiv \lambda(a_0, b_0, c_0, d_0) \pmod{N_1}$ for some scalar $\lambda \in (\mathbb{Z}/N_1\mathbb{Z})^*$.*

In the following, we provide a solution to this task. Let

$$\theta' = \lambda a_0 + N_1 a_1 + \iota(\lambda b_0 + N_1 b_1) + (\lambda c_0 + N_1 c_1 + \iota(\lambda d_0 + N_1 d_1))\pi.$$

As $\mathrm{Norm}(x + y\iota) = x^2 + y^2$, its norm equals

$$\mathrm{Norm}(\theta') = (\lambda a_0 + N_1 a_1)^2 + (\lambda b_0 + N_1 b_1)^2 + p\Big((\lambda c_0 + N_1 c_1)^2 + (\lambda d_0 + N_1 d_1)^2\Big). \quad (3.9)$$

Since $\theta \in \pi\mathbb{Z}[\iota]$ implies $a_0 = b_0 = 0$, Eq. (3.9) simplifies to

$$\mathrm{Norm}(\theta') = N_1^2(a_1^2 + b_1^2) + p\Big((\lambda c_0 + N_1 c_1)^2 + (\lambda d_0 + N_1 d_1)^2\Big). \quad (3.10)$$

Set e to be the smallest positive integer such that $eN_2/(p(c_0^2+d_0^2))$ is a quadratic residue modulo $2N_1$.

Remark 3.5.22. If N_1 were a prime, e could be chosen as the smallest quadratic non-residue modulo N_1. However, in our case N_1 is a composite number. Thus, the product of two quadratic non-residues might not be a quadratic residue if there are multiple cosets of the subgroup of quadratic residues in the group of units modulo $2N_1$.

We are primarily interested in the case where N_1 is a prime power ℓ^n. By Hensel's lemma, being a quadratic residue modulo ℓ^n is equivalent to being a quadratic residue modulo ℓ, if ℓ is odd, and equivalent to being a quadratic residue modulo 8, if $\ell = 2$.

Consequently, there is one coset of the quadratic residues in the group of units of $2N_1$ if ℓ is an odd prime. Therefore, e can be chosen to be the smallest quadratic non-residue modulo ℓ. For example, if N_1 is a power of 3 one can choose $e = 2$.

If $\ell = 2$, then there are three cosets of the quadratic residues in the group of units, i.e., the ones that contain $3, 5$, and 7 respectively. Consequently, e can always be chosen to be one of $3, 5$, or 7 in this case.

In case N_1 has distinct prime factors, for $eN_2/p(c_0^2+d_0^2)$ to be a quadratic residue it has to be a quadratic residue modulo the largest prime power dividing $2N_1$ for each distinct prime factor. If the number of cosets grows, so do the possibilities for e and thus the size of the smallest e that is guaranteed to work.

The goal is to compute θ' such that $\mathrm{Norm}(\theta') = eN_2$. Considering Eq. (3.10) modulo N_1, we obtain

$$eN_2 \equiv \lambda^2 p(c_0^2+d_0^2) \pmod{N_1}. \tag{3.11}$$

Since $eN_2/p(c_0^2+d_0^2)$ is a quadratic residue modulo $2N_1$ by the choice of e, there exists a solution for λ in Eq. (3.11) modulo $2N_1$. Compute any such solution, and lift it to the integers in $[1, 2N_1 - 1]$. Note that we do not lose generality by the lift as any other lift of λ corresponds to a change in c_1, d_1 instead.

For fixed c_0, d_0 and λ, this gives an affine relation between c_1 and d_1 modulo N_1, i.e.,

$$c_0c_1 + d_0d_1 \equiv \frac{\mathrm{Norm}(\theta') - \lambda^2 p\,(c_0^2+d_0^2)}{2\lambda p N_1} \pmod{N_1}. \tag{3.12}$$

Finally, one is left with the problem of representing an integer r as the sum of two squares, namely to find a solution (a_1, b_1) for

$$a_1^2 + b_1^2 = r := \frac{\mathrm{Norm}(\theta') - p\Big((\lambda c_0 + N_1c_1)^2 + (\lambda d_0 + N_1d_1)^2\Big)}{N_1^2} \tag{3.13}$$

where λ, c_0 and d_0 are fixed, and c_1, d_1 satisfy an affine equation modulo N_1.

The solution space to Eq. (3.12) is a translated lattice modulo N_1. More precisely, we know that c_0 or d_0 is coprime to N_1. Without loss of generality, let d_0 be coprime to N_1. Furthermore, let C denote the right hand side of Eq. (3.12). Then, (c_1, d_1) lies in the lattice

$$\langle (c_0/d_0, -1), (N_1, 0) \rangle + (C/d_0, 0). \tag{3.14}$$

Clearly, r from Eq. (3.13) can only be represented as a sum of two squares, if it is positive. This happens when the parameters N_1 and N_2 are sufficiently overstretched and imbalanced. To find a solution, one computes close vectors (c_1, d_1) to the target vector $(-\lambda c_0/N_1, -\lambda d_0/N_1)$ in the translated lattice.

Given the factorisation of r as defined in Eq. (3.13), Cornacchia's algorithm [Cor08] can then efficiently solve for a_1, b_1 or determine that no such solution exists. If no solution exists, a different vector (c_1, d_1) is chosen.

Remark 3.5.23. Cornacchia's algorithm requires the factorisation of r. This can be done in classical subexponential time or in quantum polynomial time. To avoid such computations, we apply Cornacchia's algorithm only when r is a prime and otherwise sample another close vector from the lattice.

Assuming the values of r behave like random values around pN_1^3 for the close vectors, one expects to choose $\log(pN_1^3)$ different vectors (c_1, d_1) before finding a solution for a_1, b_1 with Cornacchia's algorithm. If we do not apply Cornacchia's algorithm unless r is prime, we expect to further sample roughly $\log(pN_1^3)$ values for (c_1, d_1) until r is prime.

The volume of the translated lattice is N_1. Thus, for a generic lattice for which the Gaussian heuristic holds we expect to find a lattice point at distance N_1 from $(\lambda c_0/N_1, \lambda d_0/N_1)$. Furthermore, we can use the Hermite constant for 2-dimensional lattices to trivially bound the distance between this lattice point and the next $2\log(pN_1^3)$ closest lattice points by $\frac{8}{3}\log(pN_1^3)\sqrt{N_1}$. Thus, heuristically r is positive for the expected number of vectors (c_1, d_1) that we need to sample, whenever $eN_2 > pN_1^3 + 8/3\log(pN_1^3)\sqrt{N_1^3}$.

Remark 3.5.24. Note that for specific lattices, the Gaussian heuristic might be violated. In the worst case, we can only expect to find a lattice point at distance N_1^2 from $(\lambda c_0/N_1, \lambda d_0/N_1)$ in which case we require roughly $eN_2 > pN_1^4$.

It is easy to see that a solution for (a_1, b_1, c_1, d_1) as computed with the routine described above satisfies Eq. (3.10). The full lifting algorithm is summarised in Algorithm 3.4 and we implemented it in MAGMA [BCP97].[1]

[1]The code is available at https://github.com/SimonMerz/lifting-for-malleability-oracles

Algorithm 3.4: Lift element from $\pi\mathbb{Z}[\iota]$ to quaternion of norm N_2 or eN_2

Input: $\theta = \pi(c_0 + d_0\iota) \in \mathrm{End}(E_0)$, and parameters p, ε, N_1, N_2
Output: $\theta' = N_1a_1 + N_1b_1\iota + (\lambda c_0 + N_1c_1)\pi + (\lambda d_0 + N_1d_1)\iota\pi$ satisfying $\mathrm{Norm}(\theta') = N_2$ or eN_2 with probability $1-\varepsilon$ and $\perp$ otherwise

1 Let $e \in \mathbb{Z}_{>0}$ smallest integer s.t. $eN_2/p(c_0^2 + d_0^2) \pmod{2N_1}$ is a quadratic residue.
2 Compute λ in $eN_2 \equiv \lambda^2 p(c_0^2 + d_0^2) \pmod{2N_1}$.
3 Compute affine relation $c_0c_1 + d_0d_1 \equiv C \pmod{N_1}$.
4 Define translated lattice L containing all (c_1, d_1) satisfying the affine relation.
5 Set B to $\log(\varepsilon)\log(pN_1^3)/\log(1 - \log^{-1}(pN_1^3))$.
6 **for** $m = 1, \ldots, B$ **do**
7 Compute next closest vector (c_1, d_1) to $(-\lambda c_0/N_1, -\lambda d_0/N_1)$ in L.
8 Set r to be $\frac{\mathrm{Norm}(\theta') - p\left((\lambda c_0 + N_1c_1)^2 + (\lambda d_0 + N_1d_1)^2\right)}{N_1^2}$.
9 **if** r prime **then**
10 Use Cornacchia's algorithm to find a_1, b_1 such that $a_1^2 + b_1^2 = r$ or determine that no solution exists.
11 **if** solution found **then**
12 **return** $\theta' = N_1a_1 + N_1b_1\iota + (\lambda c_0 + N_1c_1)\pi + (\lambda d_0 + N_1d_1)\iota\pi$.
13 **return** $\perp$

An examination of Algorithm 3.4 shows that it aborts after a fixed number of trials for pairs (c_1, d_1) which leads to the following result.

Lemma 3.5.25. *Algorithm 3.4 always terminates and is correct if it returns a solution.*

We conclude this section by investigating the heuristic probability of the lifting algorithm returning a solution or aborting unsuccessfully, as well as its complexity.

Lemma 3.5.26. *Let $0 < \varepsilon < 1$. Assume r in Line 8 of Algorithm 3.4 behaves like a random value around pN_1^3. Then we expect Algorithm 3.4 heuristically to return a correct lift with probability $1-\varepsilon$ and an error $\perp$ otherwise.*

Proof. If r in Line 8 of Algorithm 3.4 behaves like a random value around pN_1^3, we expect it to be prime with probability roughly $1/\log(pN_1^3)$ and Cornacchia's algorithm to provide a solution with probability approximately $1/(\log(pN_1^3))$ due to Landau [Lan09] and Ramanujan [Ram13]. Iterating over B short vectors (c_1, d_1) of the lattice as defined in Step 6 of Algorithm 3.4, we therefore expect our algorithm to return $\perp$ with probability

$$\left(1 - \frac{1}{\log(pN_1^3)}\right)^{B/\log(pN_1^3)}.$$

Hence, iterating over $B \geq \log(\varepsilon)\log(pN_1^3)/\log(1-\log^{-1}(pN_1^3))$ as in Algorithm 3.4, we fail to find a solution with probability less than ε heuristically. □

Remark 3.5.27. In Algorithm 3.3 the lifting of endomorphisms is used for every element of the acting group G or H with cardinality $N_1/2$ and $N_1/4$, respectively. Since we expect the lifting algorithm to fail heuristically with probability ε for every single group element and the functions in Algorithm 3.3 are only exact shifts of each other when it does not fail a single time, we need to choose ε sufficiently small. Assuming independence between the different executions of the lifting algorithm, we expect to find two functions satisfying the promise of a hidden shift with probability $(1-\varepsilon)^{N_1/2} \approx 1-\varepsilon N_1/2$ by first order Taylor approximation. Thus, choosing $\varepsilon < \frac{1}{N_1}$ we expect our lifting to work with probability roughly $\frac{1}{2}$ on all endomorphisms of G and similarly $\varepsilon < \frac{2}{N_1}$ for the elements in H. By the previous lemma, the lifting remains polynomial in $\log(N_1)$ and $\log(p)$ for any such ε. Choosing ε smaller allows us to heuristically achieve a larger success probability of the algorithm. The worst-case complexity of the lifting increases linearly in $|\log(\varepsilon)|$.

Lemma 3.5.28. *Let $0 < \varepsilon < 1$. Algorithm 3.4 runs in time polynomial in $\log p$, $\log N_1$, and $|\log(\varepsilon)|$.*

Proof. The worst-case runtime of the algorithm stems from sampling B (as defined in Algorithm 3.4, Line 5) potential values of (c_1, d_1) from a lattice of dimension 2. In each iteration one needs to run a primality test, and apply Cornacchia's algorithm to a prime of size polynomial in p and N_1. □

The main drawback of our lifting algorithm is the requirement of approximately $N_2 > pN_1^3$ in case the Gaussian heuristic is satisfied for the lattice defined in Eq. (3.14), and roughly $N_2 > pN_1^4$ otherwise (see Remark 3.5.24). This bound might be partially caused by inefficiencies in the lifting algorithm. However, the following remark discusses why we can a priori not expect to find a lifting algorithm for balanced parameters that allows us to evaluate the resulting lifted endomorphism using the torsion point information provided.

Remark 3.5.29. A randomly chosen non-homogeneous quadratic equation in two variables has in general no solution. Similarly, for arbitrary endomorphisms and any N_1, N_2, we would not expect to find an endomorphism $a_1 + b_1\iota \in \mathbb{Z}[\iota]$ (in the variables a_1, b_1) inducing the same action on $\mathcal{I}$ of degree N_2. Yet, as soon as we lift an endomorphism θ to an endomorphism $\theta' = N_1(a_1 + b_1\iota + c_1\pi) + \lambda\theta$ with $c_1 \neq 0$, the degree of the lift will be of degree larger than pN_1^2. For balanced parameters we have $pN_1^2 \gg N_2$ and thus we are not able to use the provided torsion point information to evaluate the endomorphism as suggested by Algorithm 3.2.

3.5.6 Algorithm summary

We begin the summary of our attack by proving that a solution to Task 3.5.8 allows us to construct a malleability oracle for f.

Proposition 3.5.30. *Let $f_{|\mathcal{I}'} : \mathcal{I}' \to \mathfrak{J}$ be the function defined in* (3.8) *restricted to a domain $\mathcal{I}'$ so it is injective, let G be an abelian subgroup of $(\mathcal{O}_0/N_1\mathcal{O}_0)^*$ acting freely and transitively on $\mathcal{I}'$ and let $\varphi : E_0 \to E_0/K$, where $K \in \mathcal{I}'$ is chosen uniformly at random and unknown. Suppose the public parameters allow us to solve Task 3.5.8 for endomorphisms in G efficiently. Given $\varphi|_{E_0[N_2]}$, we then have a polynomial-time malleability oracle for G at $f_{|\mathcal{I}'}(K)$.*

Proof. We need to show that there exists an efficient algorithm that, on input $f(K) = f_{|\mathcal{I}'}(K) = j(E_0/K)$ and $\theta \in G$, computes $f(\theta(K))$. Let φ be the isogeny corresponding to the cyclic subgroup $K \subset E_0$ of order N_1.

The endomorphism θ has degree N_2 coprime to N_1 and using the efficient solution to Task 3.5.8, we can compute some θ' of degree N_2 such that it has the same action on the N_1-torsion as θ. Therefore, $f(\theta(K)) = E_0/\theta(K) = E_0/\theta'(K)$ up to isomorphism. By Lemma 3.5.9, this equals $(E_0/K)/\varphi(\ker\theta')$. Since $\ker\theta'$ lies in $E_0[N_2]$, we can compute its image under φ and therefore we can calculate $f(\theta(K)) = (E_0/K)/\varphi(\ker\theta')$ efficiently. □

Proposition 3.5.30 calls for solutions to the Tasks 3.5.7 and 3.5.8. In Section 3.5.4 we presented solutions to *variants* of these tasks. We use the remainder of this section to summarise the impact of these variations on the success of our approach.

Restricting the function $f : \mathcal{I} \to \mathfrak{J}$ to a subset $\mathcal{I}'$ such that $f_{|\mathcal{I}'}$ is injective and its image contains $j(E_0/K)$ for the K one aspires to recover requires information on the secret we do not posses. However, we gave three subsets $\mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3$ of $\mathcal{I}$ in Section 3.5.4 such that f restricted to any of these subsets is injective. The images of these sets under f partition all curves at distance N_1 from E_0 up to isomorphism, i.e., one of the three subsets will yield the desired result. Moreover, we provided abelian subgroups of $\mathbb{Q}[\iota] \cap (\mathcal{O}_0/N_1\mathcal{O}_0)^*$ acting freely and transitively on $\mathcal{I}_1$, $\mathcal{I}_2$, and $\mathcal{I}_3$. Thus, we can run the attack for the different $\mathcal{I}_i$ and then verify whether we have found the correct solution.

We then supply an algorithm to solve Task 3.5.20, a variant of Task 3.5.8 when N_1 and N_2 are sufficiently imbalanced, lifting endomorphisms from $\pi\mathbb{Z}[\iota]$ to ones with the same action on $\mathcal{I}$ of degree N_2 or eN_2. Here, e is a small integer depending on the parameters p, N_1, N_2 and the endomorphism. As a consequence, to use the torsion point information of $E_0[eN_2]$ under the secret isogeny given the image of $E_0[N_2]$, we need to guess the action on $E_0[e]$. Furthermore, we lift all endomorphisms in the acting group

and thus we need to guess the action on $E_0[E]$, where E is the least common multiple of *all* the e appearing for the different lifts. In Remark 3.5.22, we discuss which e might appear depending on the factorisation of N_1. For example, E is 2 if N_1 is a power of 3, or $\mathrm{lcm}(3,5,7)$ if N_1 is a power of 2. Guessing the action of the secret isogeny on $E_0[E]$ takes $O(E^3)$ trials. Finally, for efficiency reasons we lift endomorphisms from $\pi\mathbb{Z}[\iota]$, whereas the elements in the abelian groups acting on $\mathcal{I}_1$, $\mathcal{I}_2$, and $\mathcal{I}_3$ have representatives in $\mathbb{Z}[\iota]$. Further, we showed that this is no restriction via the computation of an action of the Frobenius map.

For each combination of guesses of $E_0[E]$ under the secret isogeny and whether f maps the secret K into $f(\mathcal{I}_1)$, $f(\mathcal{I}_2)$ or $f(\mathcal{I}_3)$, we can use a subexponential quantum algorithm such as Kuperberg's [Kup11] to compute the hidden shift for the functions F_K and F_J as defined in Algorithm 3.3 and verify the output of the algorithm. Both functions are injective and therefore the verification can be achieved by computing both functions on a single element and its shift respectively. Once the premise of a hidden shift is satisfied, Kuperberg's algorithm [Kup11] recovers the (correct) solution to the injective abelian hidden shift problem. Thus, we recover the secret isogeny as described in Section 3.5.3. We can summarise our result as follows.

Theorem 3.5.31. *Let $N_2 > pN_1^4$. Under the heuristics used to lift endomorphisms in Section 3.5.5, the SIDH problem can be solved in quantum subexponential time via a reduction to the injective abelian hidden shift problem.*

During this section, we have made some restrictions to simplify the presentation of our cryptanalysis.

We assumed the starting curve E_0 to be a supersingular curve with j-invariant 1728. However, the attack also applies to other curves with known endomorphism rings that are close enough to E_0 that the problem can be translated there. In Section 3.5.4, we described the required group action on $\mathcal{I}$ under the further assumption that N_1 is a power of 2, which can be generalised to powers of small primes. A sketch for powers of 3 can be found in [KMPW21, Appx. B]. Finally, we assumed that $N_1^2 < \frac{p+1}{4}$ in Lemma 3.5.11. However, to run our attack we can slightly ease this restriction. Namely, if $N_1^2 > \frac{p+1}{4}$, then we choose a divisor N_1' of N_1 such that $N_1'^2 < \frac{p+1}{4}$ and run the attack with N_1' instead. This will reveal the N_1'-part of the isogeny and then we can guess the remaining part. For sufficiently small $\frac{N_1}{N_1'}$, this may be a feasible method.

More generally, if the parameters are not quite imbalanced enough, one can always combine our attack with a step that guesses part of the secret isogeny (in exponential time). Or alternatively, if for some reason part of the secret has been leaked, and we are only interested in the remaining part of the secret. If the most significant k bits are

leaked/guessed, corresponding to the last steps of the secret isogeny, one can just run the same attack on $N_1/2^k$, as long as the parameters are imbalanced with $N_2 > p(N_1/2^k)^4$. On the other hand, if the least significant k bits are leaked, one can consider the action of the smaller subgroup $G' \subset G$ consisting of $\{a + b\iota \mid a \text{ odd}, b \text{ divisible by } 2^k\}$, where again we identify two endomorphisms with each other if they differ by multiplication by an odd scalar modulo N_1. The rest follows analogously.

3.5.7 Childs–Jao–Soukharev attack on HHS

We recall how the algorithm proposed by Childs, Jao and Soukharev [CJS14] succeeds to construct a horizontal isogeny between two given ordinary elliptic curves in quantum subexponential time (or similarly to retrieve a horizontal isogeny between oriented elliptic curves such as the secret in CSIDH [CLM+18]). We phrase their attack such that it fits into our framework using malleability oracles.

Let $\mathcal{O}$ be an order in an imaginary quadratic field. Recall from Section 2.2.5 that we have a (free) group action of the class group $\mathrm{Cl}(\mathcal{O})$ on the set of primitively $\mathcal{O}$-oriented elliptic curves which can be computed via isogenies. Recall that the hard problem is the following: given two curves connected by one (horizontal) isogeny, recover the horizontal isogeny connecting them, or equivalently invert the action by the class group. Restricting the class group action to one orbit $\mathrm{Ell}_{\overline{k}}(\mathcal{O})$ if necessary, we can further assume that this action is transitive. Childs, Jao and Soukharev provide an algorithm that constructs the sought isogeny in quantum subexponential time [CJS14] using a reduction to the hidden shift problem. We summarise their core idea as another instance of our framework using malleability oracles.

Let $\mathcal{I} := \mathrm{Cl}(\mathcal{O})$ and $\mathfrak{J} := \mathrm{Ell}_{\overline{k}}(\mathcal{O})$ for some field k. We can look at the underlying group action as a one-way function

$$f : \mathcal{I} \to \mathfrak{J} \ , \ [x] \mapsto [x] \cdot (E_0, \iota_0).$$

Since the class group $\mathrm{Cl}(\mathcal{O})$ is a group and acts free and transitively on $\mathrm{Ell}_{\overline{k}}(\mathcal{O})$, f is malleable with respect to the class group everywhere on the image.

Given (E_0, ι_0) and $(E_1, \iota_1) = [\mathfrak{b}] \cdot (E_0, \iota_0)$, one would like to compute the preimage of f at (E_1, ι_1). Childs, Jao and Soukharev observed that the functions $F_i : \mathrm{Cl}(\mathcal{O}) \to \mathrm{Ell}_{\overline{k}}(\mathcal{O})$, $[x] \mapsto [x] \cdot (E_i, \iota_i)$ for $i = 0, 1$ are shifts of each other. Moreover, they are injective functions since the action of the class group on $\mathrm{Ell}_{\overline{k}}(\mathcal{O})$ is free and transitive. The injective abelian hidden shift problem can be solved in quantum subexponential time, which allows one to recover $[\mathfrak{b}]$ and therefore an $\mathcal{O}$-oriented isogeny $\varphi : (E_0, \iota_0) \to (E_1, \iota_1)$.

3.6 Castryck–Decru attack on SIDH

In this section, we want to briefly sketch the first efficient attack on SIDH with balanced parameters due to Castryck and Decru [CD22]. Their attack spectacularly broke the only isogeny-based submission to NIST's first post-quantum standardisation process for the recommended parameters, succeeding in a matter of minutes and hours for the different parameter levels (with a MAGMA implementation running on a single core). Following the publication of their attack, improvements of the attack and related work emerged quickly. In this section, we will sketch the attack as first described by Castryck and Decru. At the end of the section, we will survey some of the related work. This section is only included for the sake of completeness and we do not claim any contribution ourselves. Rather than providing all the necessary mathematical background, we will only sketch the core idea and refer to the relevant literature instead.

In this thesis, we have only discussed isogenies between elliptic curves, which are one-dimensional abelian varieties. Similarly, one can consider isogenies between varieties of larger dimension. The core ingredients of the attack due to Castryck and Decru is to move the SSI-T problem to a decisional problem for abelian varieties of dimension 2, which can be solved by invoking a theorem due to Kani [Kan97, Thm. 2.6].

Consider the following setup for SIDH with Alice's and Bob's secret isogenies being of degree $N_1 = 2^{e_A}$ and $N_2 = 3^{e_B}$, respectively, as is usually suggested for two integers $e_A, e_B \in \mathbb{Z}$. For ease of exposition, assume $N_1 - N_2 > 0$. Bob's public key consists of the curve E_B together with the torsion point images $\varphi_B(P_A), \varphi_B(Q_A)$, where $\langle P_A, Q_A \rangle = E_0[3^{e_B}]$. Here, we use the same notation as when we introduced SIDH in Section 2.3.2. To recover Bob's secret isogeny, the attack by Castryck and Decru proceeds as follows.

First, the attacker computes any isogeny $\gamma : E_0 \to C$ of degree $N_1 - N_2$. Then, the $(2^{e_A}, 2^{e_A})$-isogeny from the two-dimensional abelian variety $C \times E_B$, that is the product of two elliptic curves, with kernel $((\gamma(P_A), \varphi_B(P_A)), (\gamma(Q_A), \varphi_B(Q_A)))$ lands again on a product of two elliptic curves by Kani's theorem [Kan97, Thm. 2.6]. However, only every roughly $10/p$-th vertex in the $(2,2)$-isogeny graph between genus 2 surfaces is a product of elliptic curves. This allows to build a distinguishing oracle which reveals (with high probability) whether a public key is valid.

Note that similarly to isogenies between elliptic curves, the $(2^{e_A}, 2^{e_A})$-isogenies between genus 2 curves can be decomposed into $(2,2)$-isogeny steps. The steps between Jacobians of genus 2, which will likely be the case for all steps of the isogeny above apart from the first and last steps, are Richelot isogenies which can be computed efficiently. For explicit formulas, we refer to Smith's thesis [Smi05, Ch. 8]. For the first step, which "glues" the

product of elliptic curves $C \times E_B$ into a genus 2 Jacobian, we refer to the formulas given in [CD22]. To verify whether a public key is valid, one just needs to check in the final step whether it is a Richelot isogeny, or whether the codomain is a product of two elliptic curves.

Given this oracle to verify whether a public key is valid, an attacker can proceed with a simple search to decision reduction, i.e. iteratively parts of Bob's secret isogeny are guessed and then verified using the oracle.

Related work. In the original attack by Castryck and Decru, the bottleneck was computing the isogeny γ for each verification of the oracle. While very efficient for suggested parameters, the asymptotic complexity was still subexponential. Using work we will present in Chapter 5, Wesolowski showed how to compute the isogeny γ in polynomial time as long as the endomorphism ring of the starting curve is known [Wes22c].

Maino and Martindale [MM22] had independently worked on breaking SIDH in a way similar to Castryck and Decru. The resulting attack uses the theory developed by Kani to directly compute private SIDH keys without using a decision to search reduction which further accelerates the key recovery. Moreover, the attack and its presentation are more reminiscent of previous torsion point attacks described earlier in this chapter.

Since all of the polynomial-time attacks mentioned so far use the knowledge of the endomorphism ring of the starting curve, hope to salvage SIDH by instantiating it with a trusted starting curve with unknown endomorphism ring existed for a brief period of time. Yet, Robert showed how to get rid of all assumptions on the endomorphism ring by moving to even larger dimensions [Rob22a].

Finally, we want to emphasise that all of the attacks on SIDH referred to in this section rely on the knowledge of torsion point images. The pure isogeny problem is not weakened by any of the attacks outlined in this section. Further, the attack requires knowledge of the degree of the sought isogeny. To this end, it was suggested to mask the degree or the torsion point images provided [Mor22, Fou22]. Unfortunately, the resulting SIDH variants are not as efficient as the original scheme.

An interesting open problem is how to use the tools provided by Kani (or recent SIDH attacks) constructively in isogeny-based cryptography. For example, Petit's torsion point attacks gave rise to a new family of trapdoor one-way functions that can be used for encryption [DDF+21], and it would be interesting to find out whether one can similarly obtain new cryptographic constructions from recent attacks. Further, can the tools developed for the attacks be used to increase efficiency in isogeny-based constructions that have not been broken such as SQISign [DKL+20]?

CHAPTER 4

Two More One-More Assumptions

The content of this chapter is based on the following two publications cryptanalysing multiple 'one-more' hardness assumptions that were used in the construction of undeniable signatures and oblivious pseudorandom functions

- Simon-Philipp Merz, Romy Minko, and Christophe Petit. Another look at some isogeny hardness assumptions. In Stanislaw Jarecki, editor, *CT-RSA 2020*, volume 12006 of *LNCS*, pages 496–511. Springer, Heidelberg, February 2020.
- Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Antonio Sanso. Cryptanalysis of an oblivious PRF from supersingular isogenies. In Mehdi

Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part I*, volume 13090 of *LNCS*, pages 160–184. Springer, Heidelberg, December 2021.

4.1 Introduction

In Section 2.4, we introduced several problems underlying isogeny-based cryptography. To construct new protocols and "prove" their security, authors often introduce tweaked hardness assumptions and conjecture the corresponding problems to be hard too. However, as this process may introduce weaknesses [KM07], it is important to scrutinise these new hardness assumptions.

In this chapter, we present multiple attacks on isogeny-based "one-more" hardness assumptions that were used in the security proofs of isogeny-based undeniable signatures and oblivious pseudorandom functions.

First, we cryptanalyse the so-called One-Sided Modified SSCDH problem and the One-More SSCDH problem underlying the security proofs of isogeny-based undeniable signature schemes proposed by Jao and Soukharev [JS14]. We show that both the decisional and computational variants of these problems can be solved in polynomial time. Further, we demonstrate an exponential attack breaking the suggested parameter sets for two undeniable signature schemes.

In the second part of this chapter, we cryptanalyse the SIDH-based oblivious pseudorandom function from supersingular isogenies proposed at Asiacrypt'20 by Boneh, Kogan and Woo [BKW20]. To account for our attack against undeniable signatures mentioned above, larger parameters were chosen. However, to prove the security of their scheme they introduce yet another tweaked assumption called the *auxiliary one-more assumption*. We give an attack on this assumption and we show that this leads to an attack on the oblivious PRF itself. The attack breaks the pseudorandomness of the protocol as it allows adversaries to evaluate the OPRF without further interactions with the server after some initial OPRF evaluations and offline computation. More specifically, we first propose a polynomial-time attack. Then, we argue that it is easy to change the OPRF protocol to include some countermeasures, and present a second subexponential attack that succeeds in the presence of said countermeasures. Both attacks break the security parameters suggested by Boneh et al. which we demonstrate in practice using a proof of concept implementation of our attack. Finally, we examine the generation of one of the OPRF parameters and argue that a trusted third party is needed.

Chapter outline. In Section 4.2.1, we recall the One-Sided Modified SSCDH problem and the One-More SSCDH problem, two problems that were conjectured to be hard in the literature on isogeny-based undeniable signatures. We describe an attack on both of them in Section 4.2.2. In the following Section 4.2.3, we describe how Jao and Soukharev used the problems to construct isogeny-based undeniable signatures [JS14]. We provide an attack on the signature scheme itself and mention further constructions that are affected by our attacks in Section 4.2.4.

In Section 4.3.1, we give a brief introduction to OPRFs and briefly survey some use cases of this protocol in practice, before we recall the security properties of (verifiable) OPRFs in Section 4.3.2 more formally. In Section 4.3.3, we describe the construction due to Boneh, Kogan and Woo. The attacks against their new "one-more" assumption are presented in Section 4.3.5 and analysed in Section 4.3.6. In Section 4.3.7, we discuss how to apply the attack against the OPRF protocol itself. We present experimental results of our attack's proof of concept implementation in Section 4.3.8. In Section 4.3.9, we argue that a trusted setup should be used to generate one the OPRF's parameters and briefly sketch two pitfalls in case of a lack of such a trusted setup.

4.2 Cryptanalysis of undeniable signatures based on SIDH

In this section, we review some of the isogeny problems that have been suggested as hard problems to prove the security of a new construction of isogeny-based undeniable signatures by Jao and Soukharev [JS14].

An undeniable signature scheme is a scheme in which signatures can only be verified with cooperation of the signer [CV90]. To verify a signature, the verifier sends a signature σ back to the signer, who engages in a zero-knowledge confirmation (or disavowal) protocol to prove the validity (or invalidity) of σ. The security properties required by an undeniable signature scheme are undeniability, unforgeability and invisibility. Undeniability ensures that a signer cannot repudiate a valid signature. Unforgeability is the notion that an adversary cannot compute a valid message-signature pair without knowledge of the signer's secret key. Invisibility requires that an adversary cannot distinguish between a valid signature and a signature produced by a simulator with non-negligible probability. For more background on undeniable signature schemes we refer the reader to [CV90, DP96, KF08].

The construction of undeniable signatures by Jao and Soukharev has been used and extended by other authors (e.g. [SC18]). We will show that the hardness assumptions

used to make the security proofs work are not valid and that the proposed isogeny problems lack the conjectured hardness. This does not immediately lead to an attack on the signature scheme itself. However, we propose an (exponential) attack on the cryptographic construction, breaking the suggested parameters for all security levels.

4.2.1 Modified supersingular CDH problems

In this section, we recall the somewhat more artificial variations of the supersingular computational Diffie–Hellman problem underlying the SIDH key exchange. These variations were used and conjectured to be hard in the security proofs of [JS14, SC18].

In the following, we use the same notation as in Section 2.3.2, where we introduced SIDH, with $\deg(\varphi_A) = \ell_A^{e_A}$ and $\deg(\varphi_B) = \ell_B^{e_B}$ for small distinct primes ℓ_A, ℓ_B.

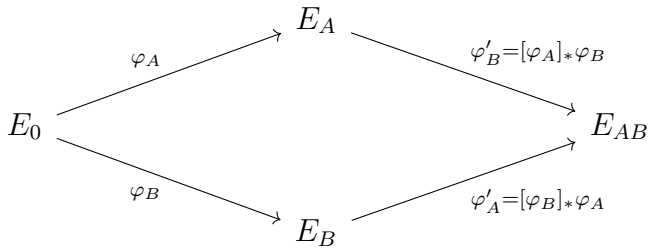

Fig. 4.1 Commutative SIDH diagram.

Definition 4.2.1 ([JS14])**.** Fix the notation as in Fig. 4.1. Given E_A, E_B and $\ker(\varphi_B)$, the *modified SSCDH (MSSCDH) problem* asks to determine E_{AB} up to isomorphism.

Clearly, the knowledge of $\ker(\varphi_B)$ is equivalent to the knowledge of $\varphi_B : E_0 \to E_B$, but the lack of information on the auxiliary points in the image of φ_A in the MSSCDH problem compared to the SSI-T problem underlying SIDH prevents an attacker to *shift* $\ker(\varphi_B)$ onto E_A to compute the pushfoward $\varphi'_B = [\varphi_A]_*\varphi_B : E_A \to E_{AB}$.

Definition 4.2.2 ([JS14])**.** For fixed E_A, E_B, given an oracle to solve MSSCDH for any E_A, $E_{B'}$, $\ker(\varphi_{B'})$, where $E_{B'}$ is $\ell_B^{e_B}$-isogenous to E_0 and not isomorphic to E_B, the *one-sided MSSCDH (OMSSCDH) problem* asks to solve MSSCDH for E_A, E_B and $\ker(\varphi_B)$.

While the OMSSCDH assumption seems somewhat more artificial, it arises naturally in the security analysis of undeniable signatures proposed in [JS14]. The authors of the same paper conjectured the problem to be computationally infeasible, in the sense that for any polynomial-time algorithm, the advantage of the algorithm is a negligible function in the security parameter $\log p$. However, we will see in the next subsection that

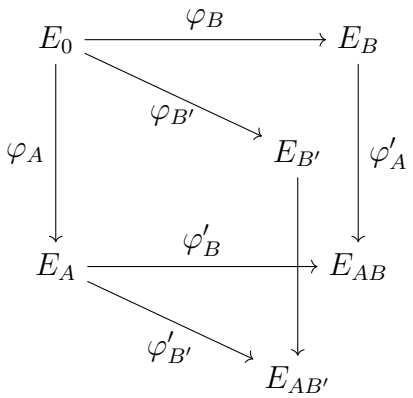

Fig. 4.2 Given an oracle providing $E_{AB'}$ for any curve E'_B not isomorphic to E_B that is $\ell_B^{e_B}$-isogenous to E_0, the OMSSCDH asks to find E_{AB}.

a polynomial time attacker will have a non-negligible advantage to solve the OMSSCDH problem. The decisional variant of this problem is also defined in [JS14]; our attack can be applied to it in a straightforward way.

Our results furthermore break other strongly related problems, such as the following slightly weaker problem used in the construction of undeniable blind signatures [SC18].

Definition 4.2.3. Let E_0 be some starting curve as in the SIDH key exchange and let m_A, n_A be secret integers in $\{0, \ldots, \ell_A^{e_A} - 1\}$.

Let a signing oracle respond $E_{AB} \cong E_B/\langle [m_A]P_B + [n_A]Q_B \rangle$ upon receipt of a curve E_B isogenous to E_0 and points P_B, Q_B spanning $E_B[\ell_B^{e_B}]$.

The *one-more SSCDH (1MSSCDH) problem* asks to produce at least $q+1$ distinct pairs of curves (E_{B_i}, E_{AB_i}), where E_{B_i} are $\ell_B^{e_B}$-isogenous to E_0, P_{B_i} and Q_{B_i} span $E_{B_i}[\ell_B^{e_B}]$ and E_{AB_i} is isomorphic to $E_{B_i}/\langle [m_A]P_{B_i} + [n_A]Q_{B_i} \rangle$ for $1 \leq i \leq q+1$, after q queries to the signing oracle.

Compared to the OMSSCDH problem, 1MSSCDH leaves the choice of the additional MSSCDH instance which needs to be solved to the attacker.

4.2.2 Attacking OMSSCDH and 1MSSCDH

Now, we describe our attacks on the OMSSCDH and 1MSSCDH problems.

Proposition 4.2.4. *A solution to the OMSSCDH problem can be guessed with probability* $\frac{1}{(\ell_B+1)\ell_B}$ *after a single query to the signing oracle.*

Proof. Assume an attacker wants to solve OMSSCDH given E_A, E_B and $\ker(\varphi_B)$. Let $E_{B'}$ be another curve ℓ_B^2-isogenous to E_B and $\ell_B^{e_B}$-isogenous to E_0. Such an $E_{B'}$ is

obtained from E_B by backtracking the last ℓ_B-isogeny step of φ_B and then computing another ℓ_B isogeny from this curve that does not lie on the path of φ_B. Since φ_B is known this can be computed without any guessing, but even if φ_B was not known to the attacker such an $E_{B'}$ could be guessed with probability $\frac{\ell_B - 1}{(\ell_B+1)\ell_B}$.

Next, the attacker queries the oracle on $E_{B'}$ to receive $E_{AB'}$. As depicted in Fig. 4.3, any curve in the isomorphism class of E_{AB} is ℓ_B^2-isogenous to $E_{AB'}$. Therefore, an attacker guesses the isomorphism class of E_{AB} correctly with probability $((\ell_B + 1)\ell_B)^{-1}$ finishing the proof. □

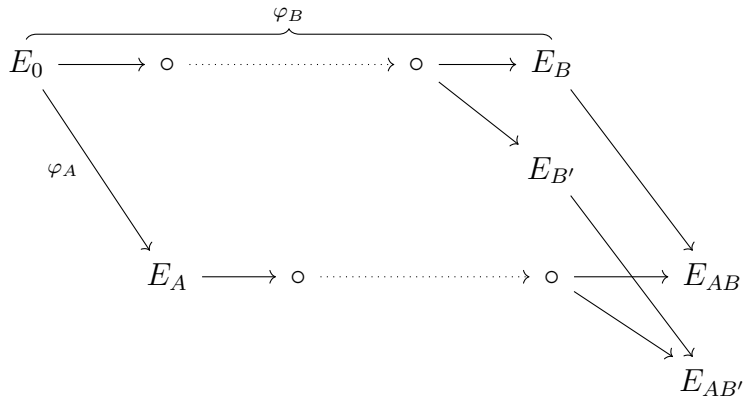

Fig. 4.3 Query of OMSSCDH oracle on ℓ_B^2-isogenous curve via backtracking one step of φ_B yields elliptic curve ℓ_B^2-isogenous to target curve

To compute φ_B efficiently, the prime ℓ_B is usually chosen to be small (typically 2 or 3) and thus Proposition 4.2.4 breaks the OMSSCDH problem completely.

Remark 4.2.5. Using multiple queries to the signing oracle, one can break the OMSSCDH problem with even larger probability. For example, querying the oracle on two curves ℓ_B^2-isogenous to E_B and $\ell_B^{e_B}$-isogenous to E_0, the common neighbour of both answers of the oracle will be ℓ_B isogenous to E_{AB} and thus can be guessed with probability $(\ell_B - 1)^{-1}$.

Without the condition on the degree of the isogeny between the curves submitted to the OMSSCDH oracle and the base curve (or if this condition is not checked), the attack can be made even more efficient. Namely, an attacker always solves this modified version of the OMSSCDH problem after two queries to the oracle as follows.

The attacker computes two curves E_{B_1}, E_{B_2} of different isomorphism classes that are ℓ_B-isogenous to E_B. Knowing $\ker(\varphi_B)$ the attacker computes $\ker(\varphi_{B_i})$ and queries the oracle to solve MSSCDH for E_A, E_{B_i} and $\ker(\varphi_{B_i})$ for $i = 1, 2$. The oracle sends back

E_{AB_i} which are ℓ_B-isogenous to the unknown E_{AB} as depicted in Fig. 4.4. We find the isomorphism class of E_{AB} as the only common ℓ_B-isogenous neighbour to both E_{AB_1} and E_{AB_2}.

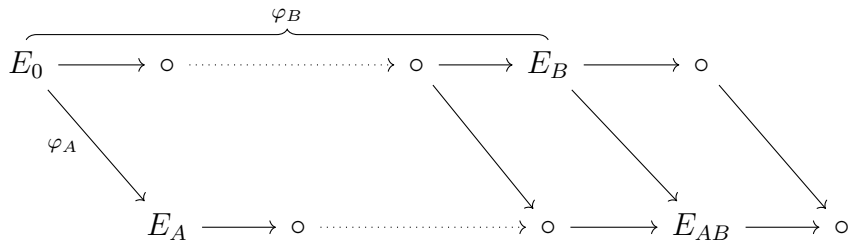

Fig. 4.4 Diagonal maps show the signing oracle mapping ℓ_B-isogenous curves of E_B to ℓ_B-isogenous curves of the target curve E_{AB}

Clearly, the attack described in Proposition 4.2.4 applies to the decisional version of the problem, OMSSDDH, too. Furthermore, a solution to the OMSSCDH problem implies a solution to the 1MSSCDH problem.

Corollary 4.2.6. *A solution to the 1MSSCDH problem can be guessed with probability* $\frac{1}{(\ell_B+1)\ell_B}$ *after a single query to the signing oracle.*

Note that alternatively an attacker could submit all of the ℓ_B^2-isogenous neighbours via backtracking one step of φ_B and compute "one-more" solution as the only remaining option with certainty.

4.2.3 Application to the construction by Jao and Soukharev

We continue with a description of our attack against the isogeny-based undeniable signature scheme by Jao and Soukharev [JS14].

Jao–Soukharev undeniable signatures

The protocol by Jao and Soukharev takes p to be a prime of the form $\ell_A^{e_A}\ell_B^{e_B}\ell_C^{e_C} \cdot f \pm 1$ similar to the SIDH protocol, where ℓ_A, ℓ_B, ℓ_C are small coprime primes. A supersingular starting curve E_0 over $\mathbb{F}_{p^2}$ and bases $\{P_A, Q_A\}$, $\{P_B, Q_B\}$ and $\{P_C, Q_C\}$ of $E_0[\ell_A^{e_A}]$, $E_0[\ell_B^{e_B}]$ and $E_0[\ell_C^{e_C}]$, respectively, are fixed. The public parameters of the scheme are p, E_0 and the three torsion bases, together with a hash function H. The signer generates random integers $m_A, n_A \in \mathbb{Z}/\ell_A^{e_A}\mathbb{Z}$ and computes the isogeny $\varphi_A : E_0 \to E_A := E_0/\langle [m_A]P_A + [n_A]Q_A \rangle$. The public key consists of the curve E_A together with the points $\{\varphi_A(P_C), \varphi_A(Q_C)\}$ and the integers m_A, n_A, or equivalently φ_A, constitute the private key.

To sign a message M, the signer computes the hash $h = H(M)$ of the message and the isogenies

$$\begin{aligned}\varphi_B &: E_0 \to E_B = E_0/\langle P_B + [h]Q_B\rangle \\ \varphi_{AB} &: E_A \to E_{AB} = E_A/\langle \varphi_A(P_B + [h]Q_B)\rangle \\ \varphi_{BA} &: E_B \to E_{AB} = E_B/\langle \varphi_B([m_A]P_A + [n_A]Q_A)\rangle.\end{aligned}$$

The signer outputs E_{AB} in addition to the two auxiliary points $\varphi_{BA}(\varphi_B(P_C)), \varphi_{BA}(\varphi_B(Q_C))$ as the signature.

Given a signature $\sigma = (E_\sigma, P, Q)$, the first step in the confirmation and disavowal protocols is for the signer to select $m_C, n_C \in \mathbb{Z}/\ell_C^{e_C}\mathbb{Z}$ and compute the curves $E_C = E_0/\langle [m_C]P_C + [n_C]Q_C\rangle$, $E_{BC} = E_B/\langle \varphi_B([m_C]P_C + [n_C]Q_C)\rangle$, $E_{AC} = E_A/\langle \varphi_A([m_C]P_C + [n_C]Q_C)\rangle$ and $E_{ABC} = E_{BC}/\langle \varphi_B([m_A]P_A + [n_A]Q_A)\rangle$. The signer outputs these curves and $\ker(\varphi_{CB})$ as their commitment, where φ_{CB} is the isogeny from E_C to E_{BC}. In addition to the auxiliary points of the signature, this commitment provides the verifier with enough information to compute E_{ABC} and $E_{\sigma C} = E_\sigma/\langle [m_C]P + [n_C]Q\rangle$, to check whether $E_{\sigma C} = E_{ABC}$. Further details of the confirmation and disavowal protocols can be found in [JS14].

In the Jao–Soukharev construction, the adversary knows E_A and can compute E_{B_i} and $\ker(\varphi_{B_i})$, corresponding to message M_i, from H. The signing oracle then essentially solves MSSCDH for any of the adversary's input messages M_i. The paper claims that under the assumption that the confirmation and disavowal protocols of the signature scheme are zero-knowledge, the *unforgeability game* describes the OMSSCDH problem. We first recall the unforgeability game and then argue that OMSSCDH is not equivalent to forging signatures in the next subsection.

Unforgeability is the notion that an adversary cannot compute a valid message-signature pair with non-negligible probability. It is defined using the following security game:

1. The challenger generates a key pair, giving the verification key to the adversary.

2. The adversary is given access to a signing oracle and makes queries adaptively with messages m_i, for $i = 1, 2, \ldots, k$, for some k, receiving the corresponding signatures σ_i.

   Additionally, the adversary has access to a confirmation/disavowal oracle for the protocol, which they can query adaptively with message-signature pairs.

3. The adversary outputs a pair (m, σ).

The adversary wins the game (i.e. forges a signature successfully), if (m, σ) is a valid message-signature pair and $m \neq m_i$ for all $i = 1, 2, \ldots k$. A signature scheme is called *unforgeable* if any PPT adversary wins with only negligible probability.

Another look at the security proof of [JS14]

In [JS14] the claim is made that forging a signature for this construction is equivalent to solving OMSSCDH, so one would expect our attack to directly break unforgeability. However, equivalence would only be true if the attacker had the freedom to submit arbitrary curves to the signing oracle. In the protocol, an adversary wishing to forge a signature can only query the signing oracle with messages, M_i. In the signing protocol the curves E_{B_i} are computed from message hashes rather than the messages themselves. Thus, an adversary would need to find a message mapping to some specific curve first in order to use an attack on OMSSCDH to forge a signature. Consequently, the adversary would need to break the hash function. Forging messages therefore seems harder than breaking OMSSCDH.

Thus, the attack of Section 4.2.1 applies to the hardness assumption but not the actual protocol in [JS14]. In this section we will demonstrate how a hybrid version of our attack on OMSSCDH and finding "near-collisions" in the hash function allows to reduce the security level of the construction for the proposed parameters.

To account for the scheme's loss of malleability due to the hash function we make use of the following lemma.

Lemma 4.2.7. *Let E_0 be a supersingular elliptic curve, let ℓ be a prime, let e be an integer, and let $\{P, Q\}$ be a basis for $E_0[\ell^e]$. Let $n, m < \ell^e$ be positive integers congruent modulo ℓ^k for some integer $k < e$. Then the ℓ-isogeny paths from E_0 to $E_A = E_0/\langle P + [n]Q\rangle$ and $E_B = E_0/\langle P + [m]Q\rangle$ are equal up to the k-th step.*

Proof. Let $m = n + \alpha\ell^k$, for some $\alpha > 0$. We have

$$\ell^{e-k}(P + [m]Q) = \ell^{e-k}(P + [n]Q) + \ell^{e-k+k}[\alpha]Q = \ell^{e-k}(P + [n]Q).$$

That is, both $\langle P + [m]Q\rangle$ and $\langle P + [n]Q\rangle$ are equal on a subgroup of size ℓ^k. Now, the lemma follows from Corollary 2.2.15. □

Suppose the adversary wishes to forge a signature for the message M. Let $H : \{0,1\}^* \to \mathbb{Z}$ be the public hash function used in the signature scheme. The hash function

determines a coefficient of a point in the $E[\ell_i^{e_i}]$ torsion group and can therefore be treated as a function to a group of size $2^{2\lambda}$ for classical security levels. Let 2^L denote the size of this group in the image.

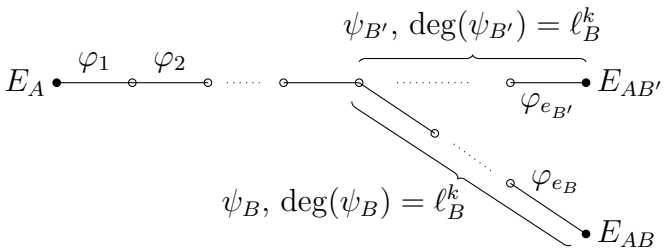

Fig. 4.5 Isogeny paths between E_A, E_{AB} and $E_{AB'}$. In our attack we use $\psi = \psi_B \circ \hat{\psi}_{B'}$ and have $\varphi_{AB'} = \varphi_{e_{B'}} \circ \varphi_{e_{B'}-1} \circ \cdots \circ \varphi_1$.

The attack proceeds as follows:

1. Build a near-collision on H with respect to the ℓ_B-adic metric. More precisely, find two messages M and M' such that the difference between $H(M)$ and $H(M')$ is divisible by a large power of ℓ_B, say a power of size roughly 2^{L_1}.

2. Submit M' to the signing oracle to obtain the signature

$$\sigma' = \Big(E_{AB'}, P_1 := \varphi_{B'A}(\varphi_{B'}(P_C)), P_2 := \varphi_{B'A}(\varphi_{B'}(Q_C))\Big).$$

3. Guess the ℓ_B^{2k}-isogeny $\psi : E_{AB'} \to E_{AB}$, where E_{AB} is the unknown curve corresponding to M. Let $\psi = \psi_B \circ \hat{\psi}_{B'}$, the composition of two degree $\ell_B^k \approx 2^{L_2}$ isogenies with $L_2 = L - L_1$, where $\hat{\psi}_{B'}$ corresponds to k backwards steps on the isogeny path from $E_{AB'}$ and ψ_B corresponds to k forward steps to E_{AB}. This is illustrated in Fig. 4.5. The probability of correctly identifying ψ in a single guess is $\frac{1}{(\ell_B+1)\ell_B^{2k-1}}$.

4. Find s such that $s\ell_B^k \equiv 1 \bmod \ell_C^{e_C}$. Compute the auxilary points of the signature as $\{[s] \cdot \psi(P_1), [s] \cdot \psi(P_2)\}$.

5. Output $\sigma = (E_{AB}, [s] \cdot \psi(P_1), [s] \cdot \psi(P_2))$.

Theorem 4.2.8. *Let s, ψ, P_1 and P_2 be defined as in our attack. Moreover, let σ be the signature $(E_{AB}, [s]\psi(P_1), [s]\psi(P_2))$ computed in the attack. Assuming that E_{AB} is guessed correctly, σ is a valid signature.*

Proof. Since ψ is of degree coprime to ℓ_C, we have that $\langle \psi(P_1), \psi(P_2) \rangle = E_{AB}[\ell_C^{e_C}]$ whenever $\langle P_1, P_2 \rangle = E_{AB'}[\ell_C^{e_C}]$. Although these points would already pass the validation

for the signature scheme, one might be able to distinguish them from the honestly generated points. The factor $[s]$ in our signature ensures that forged and honest signatures are identically distributed, as we will show below.

Recall that $\psi = \psi_B \circ \hat{\psi}_{B'}$ and $P_1 = \varphi_{B'A}(\varphi_{B'}(P_C))$. Since the order of P_C is coprime to $\deg(\varphi_{B'A})$ and $\deg(\varphi_{B'})$, and the isogeny diagram is commutative, we can write $P_1 = \varphi_{AB'}(\varphi_A(P_C))$.

By expanding $\varphi_{AB'}$ we obtain

$$\begin{aligned}\hat{\psi}_{B'} \circ \varphi_{AB'} &= \hat{\varphi}_{e_{B'}-k} \circ \cdots \circ \hat{\varphi}_{e_{B'}} \circ \varphi_{e_{B'}} \circ \cdots \circ \varphi_{e_{B'}-k} \circ \ldots \varphi_{e_B-k} \circ \cdots \circ \varphi_1 \\ &= [\ell_B^k] \circ \varphi_{e_{B'}-k-1} \circ \cdots \circ \varphi_1.\end{aligned}$$

Since s is the multiplicative inverse of ℓ_B^k modulo $\ell_C^{e_C}$, we have

$$[s]\psi(P_1) = \varphi_{AB}(\varphi_A(P_C)) \in E_{AB}[\ell_C^{e_C}].$$

Analogously, we have $[s]\psi(P_2) = \varphi_{AB}(\varphi_A(Q_C)) \in E_{AB}[\ell_C^{e_C}]$.

Let $P = \varphi_{BA}(\varphi_B(P_C)) \in E_{AB}[\ell_C^{e_C}]$ and $Q = \varphi_{BA}(\varphi_B(Q_C)) \in E_{AB}[\ell_C^{e_C}]$. These are the points we expect in an honest signature. In both the confirmation and disavowal protocols of the Jao–Soukharev scheme, the verifier uses the auxiliary points to compute an isogeny from E_{AB} to a curve E_σ defined as $E_{AB}/\langle [m_C \cdot s]\psi(P_1) + [n_C \cdot s]\psi(P_2)\rangle$, where $m_C, n_C \in \mathbb{Z}/\ell_C^{e_C}\mathbb{Z}$ are integers chosen by the signer. This curve is checked against $E_{ABC} = E_{AB}/\langle [m_C]P + [n_C]Q\rangle$ to determine the validity of σ. The two points obtained in our attack span the subgroup $E_{AB}[\ell_C^{e_C}]$, and we have E_{AB} as the correct signature curve, so it follows that E_σ and E_{ABC} are isomorphic and thus the signature is accepted as valid. □

Finding a near-collision of L_1 bits on H classically has a cost of $2^{L_1/2}$ queries to H. In Step 3, we can then guess the correct isogeny and curve E_{AB} with probability approximately $2^{-2L_2} = 2^{-2(L-L_1)}$. Taking $L_1 = 4L/5$ the attack then has a total classical cost of $2^{2L/5}$, as opposed to the claimed $2^{L/2}$. This lowers the security estimate of the parameters with respect to unforgeability. Moreover, we are able to break invisibility, since any adversary with the ability to forge signatures with higher probability can simply check whether the challenge signature obtained in the invisibility game matches a potential forgery.

Assuming that we can find (near)-collisions of the hash function with lower quantum complexity [BHT97], the first step of our attack costs $2^{L_1/3}$ on a quantum computer. Taking $L_1 = 6L/7$, this could lower the complexity on a quantum computer to $2^{2L/7}$, as

opposed to the claimed $2^{L/3}$. However, it has been argued that quantum collision search might be inferior to classical collision search because of the expensive memory access and quantum memory. For a general discussion on the impracticalities of known quantum algorithms for collision search, we refer to Bernstein [Ber09].

4.2.4 Srinath and Chandrasekaran undeniable blind signatures

Srinath and Chandrasekaran [SC18] extended the Jao–Soukharev construction to an undeniable *blind* signature scheme, introducing a third actor, the requestor, to the scheme. It is a four-prime variant of the original scheme, taking the prime p to be of the form $\ell_A^{e_A}\ell_B^{e_B}\ell_C^{e_C}\ell_D^{e_D} \cdot f \pm 1$ and adding the public parameter $\{P_D, Q_D\}$, a basis for $E_0[\ell_D^{e_D}]$. The requestor computes the message curve $E_B = E_0/\langle P_B + [H(m)]Q_B\rangle$ using the public hash function, as before. They then blind the curve by taking a random integer $0 < d < \ell_D^{e_D}$ to compute $E_{BD} = E_B/\langle \varphi_B(P_D) + [d]\varphi_B(Q_D)\rangle$. The blinded curve is then sent to the signer. The signing algorithm of the scheme proceeds in the same way as in the Jao–Soukharev construction. Upon receipt of the blinded signature curve E_{BDA}, the requestor uses an unblinding algorithm to obtain the unblinded signature E_{BA}, which is the same as the one in Jao–Soukharev's signature scheme. Thus, signatures as in Srinath and Chandrasekaran are just Jao–Soukharev signatures pushed forward through another coprime isogeny and the scheme is vulnerable to our attack. As before, both unforgeability and invisibility can be broken.

4.3 Cryptanalysis of an oblivious PRF from supersingular isogenies

Boneh, Kogan and Woo proposed two new post-quantum oblivious pseudorandom functions (OPRFs) at Asiacrypt 2020 [BKW20]. One construction is based on SIDH and the other one on CSIDH. The security of the SIDH-based variant relies, amongst other assumptions, on the hardness of a new "one-more" assumption that we cryptanalyse in this section. We will provide multiple variants of an attack on the assumption and the OPRF protocol. The CSIDH-based OPRF proposal by Boneh, Kogan and Woo is not affected by our attacks.

Before presenting the attacks, we recall what an OPRF protocol is, point to different post-quantum constructions and recall the required security properties.

The attacks on the SIDH-based OPRF break the pseudorandomness property of an OPRF and allow malicious clients to evaluate the OPRF on arbitrary inputs after some initial queries to the server, without further interaction with the server.

Our first attack allows attackers to break this property of the SIDH-based OPRF protocol in polynomial time. We argue that a simple modification of the OPRF protocol prevents such an attack. Then, we show that a second variant of the attack leads to an attack on the protocol even in the presence of those countermeasures. The latter attack has a subexponential complexity, but there appear to be no simple countermeasures. As a result of our attack, the parameters suggested by Boneh, Kogan and Woo fall short of their estimated security level.

Finally, we discuss which party should generate one of the parameters of the SIDH-based OPRF. We argue there are security implications if the server, the client or any third party maliciously generates this parameter. The client or a third party could introduce a backdoor through this parameter to recover the secret key of the server, whereas if the server is malicious, they can break another assumption on which the security proofs are built. We suggest that a trusted setup may be needed to guarantee provable security.

4.3.1 OPRFs and their applications

An oblivious pseudorandom function (OPRF) is a two-party protocol between a client and a server that computes a pseudorandom function (PRF) on a client's input with the server's key. At the end, the server does not learn anything about the client's input or the output of the function and the client learns the evaluation of the OPRF but nothing about the server's key. In particular, a client should not be able to compute the OPRF on any input without the server's participation.

Moreover, a *verifiable* oblivious pseudo random function (VOPRF) is an OPRF where a server initially commits to some key and the client is ensured that the server used this key to evaluate the OPRF consistently. In particular, the client is guaranteed that a server does not change their secret key across different executions of the protocol.

Oblivious pseudorandom functions are an important building block in many cryptographic applications. They can be used for private set intersection [JL09], which in turn has many applications such as private contact discovery for messaging services [DRRT18] or checking for compromised credentials [LPA+19]. Further applications of (V)OPRFs include password-authenticated key exchange [JKX18], password-management systems [ECS+15], adaptive oblivious transfer [JL09], password-protected secret sharing [JKK14] and privacy-preserving CAPTCHA systems [DGS+18].

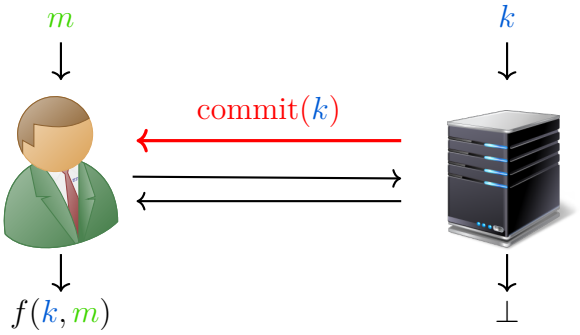

Fig. 4.6 (V)OPRF protocol between a client and a server.

Apart from their theoretical relevance in cryptography, OPRFs have had significant real-world impact recently.

The password-authenticated key exchange OPAQUE [JKX18] which is built on an OPRF is intended for use in TLS 1.3 [SKFB21].

The privacy-preserving authorisation mechanism known as Privacy Pass by Davidson, Goldberg, Sullivan, Tankersley and Valsorda [DGS+18] is also based entirely on the security of a VOPRF. Privacy Pass is currently used at scale by Cloudflare. Finally, there is an ongoing effort to standardise OPRFs at the Crypto Forum Research Group (CFRG) [DSW19].

To construct verifiable OPRFs, generic techniques from two-party computation and zero-knowledge proofs can be used. However, the resulting protocols might be rather inefficient. Therefore, all of the real-world use cases of (V)OPRFs are currently instantiated with efficient (V)OPRFs which are based on classical security assumptions. Practical constructions are currently usually based either on the hardness of the decisional Diffie–Hellman problem, called DH-OPRF [JKK14], or they are derived from RSA blind signatures [Cha82, DSW19].

For quantum-secure OPRFs, there are only few proposals. Indeed, only three such solutions appear in the literature to date. In 2019, Albrecht, Davidson, Deo and Smart proposed a lattice-based VOPRF [ADDS21] based on the ring learning with errors problem and the short integer solution problem in one dimension. Another OPRF based on the shifted Legendre symbol problem was proposed in [SHB21] and Boneh, Kogan and Woo presented two isogeny-based (V)OPRFs at Asiacrypt 2020 [BKW20]. One construction is an SIDH-based VOPRF, and the other a CSIDH-based OPRF.

The SIDH-based variant relies on the hardness of SIDH, and a novel "one-more" isogeny assumption which we attack in Section 4.3.5. Our attack on the assumption further breaks the pseudorandomness of the OPRF.

4.3.2 Security properties of (V)OPRFs

The security properties of an oblivious pseudorandom function (OPRF) include those of a standard pseudorandom function (PRF), see e.g. [KL20, Def. 3.25].

Definition 4.3.1. Let $F : K \times X \to Y$ be an efficiently computable function. F is a *pseudorandom function* (PRF) if for all probabilistic polynomial-time distinguishers D, there is a negligible function negl such that

$$\Pr[D^{F(k,\cdot)}(1^n) = 1] - \Pr[D^{f(\cdot)}(1^n) = 1] \leq \mathsf{negl}(n),$$

where Pr denotes the probability, the first probability is taken over uniform choices of $k \in \{0,1\}^n$ and the randomness of D, and the second probability is taken over uniform choices of functions $f : X \to Y$ and the randomness of D.

A consequence of pseudorandomness is that one cannot compute evaluations of $F(k,\cdot)$ on new inputs from existing evaluations. However, our attack on the OPRF by Boneh, Kogan and Woo will allow adversaries to evaluate $F(k,\cdot)$ on arbitrary inputs after some initial evaluations. This could lead to significant attacks on OPRF-based protocols. In the context of private set intersection based on oblivious PRFs, the proposed attack allows the attacker to brute-force the other party's set elements and break the privacy requirement. In the Privacy Pass protocol used to guarantee privacy-preserving CAPTCHAs, our attack allows the attacker to generate unlimited tokens, thus avoiding solving CAPTCHAs and fully breaking the security of the system.

Furthermore, OPRFs are required to be *oblivious* in the following sense.

Definition 4.3.2 ([FIPR05])**.** Let $F : K \times X \to Y$ be a PRF. A protocol between a client with input $x \in X$ and a server with key $k \in K$ is called an *oblivious* PRF, if the client learns $F(k,x)$ and nothing else and the server learns nothing about x or $F(k,x)$ at the end of the protocol.

In particular, the server will learn nothing about the input x of the client and the client will learn nothing about the server's key k.

Additionally, an OPRF can have the property of being verifiable.

Definition 4.3.3. An OPRF is said to be *verifiable* if the evaluation y that the client obtains at the end of the protocol is correct, i.e. if it satisfies $y = F(k,x)$, where $x \in X$ is the client's input and $k \in K$ is the server's private key.

In practice, verifiability is usually ensured by the server committing to a key k prior to the execution of the verifiable OPRF (VOPRF) and providing a zero-knowledge proof that the VOPRF execution uses the same key as the committed value.

For more formal simulation-based security definitions of properties required from OPRFs, we refer to [ADDS21].

4.3.3 An isogeny-based OPRF by Boneh, Kogan and Woo

We provide a simplified description of the SIDH-based OPRF by Boneh, Kogan and Woo which we sketch in Fig. 4.7.

Let λ be the security parameter and let $p = fN_{\mathsf{K}}N_{\mathsf{M}}N_{\mathsf{V}}N_{\mathsf{R}}N_{\mathsf{S}} - 1$ be a prime where $f \in \mathbb{Z}$ is a small cofactor and N_i are powers of distinct small primes such that $N_{\mathsf{K}}, N_{\mathsf{M}}, N_{\mathsf{V}}, N_{\mathsf{R}}$ are roughly of size $2^{5\lambda/2}$ and $N_{\mathsf{S}} \approx 2^{2\lambda}$. To account for our attack from [MMP20] described in Section 4.2.3, the factors $N_{\mathsf{K}}, N_{\mathsf{M}}, N_{\mathsf{V}}, N_{\mathsf{R}}$ are chosen of size $2^{5\lambda/2}$ instead of the more common size $2^{2\lambda}$ in the SIDH protocol. Moreover, let $H_1 : \{0,1\}^* \to \mathbb{Z}_{N_{\mathsf{M}}}$ be a cryptographic hash function. In the security proofs, H_1 is treated as a random oracle. Finally, let E_0 be a randomly chosen supersingular elliptic curve over $\mathbb{F}_{p^2}$ and let $\{P_i, Q_i\}$ denote a basis of $E_0[N_i]$ for $i = K, M, V, R, S$. While Boneh, Kogan and Woo only require E_0 to be a randomly chosen elliptic curve, we will discuss how it is generated in Section 4.3.9 and argue that this choice should be done by a trusted third party.

First, the server chooses their private key k which is the PRF key and publishes a commitment to this key. To evaluate the OPRF on the input x in the input space, a client computes the hash $H_1(x) = m \in \mathbb{Z}_{N_{\mathsf{M}}}$. Furthermore, the client randomly chooses an element $r \in \mathbb{Z}_{N_{\mathsf{R}}}$.

The client computes the isogenies $\phi_m : E_0 \to E_m := E_0/\langle P_M + [m]Q_M\rangle$ and $\phi_r : E_m \to E_{mr} := E_m/\langle \phi_m(P_R) + [r]\phi_m(Q_R)\rangle$. Then, the client sends E_{mr} together with the torsion point images of P_i, Q_i for $i = V, K, S$ to the server as well as a basis of $E_{mr}[N_{\mathsf{R}}]$. To avoid active attacks like the GPST attack [GPST16], where a malicious client tries to recover information about the server's private key by sending manipulated torsion point information, the client proves to the server in a non-interactive zero-knowledge proof that they know the kernel of the isogeny from E_0 to E_{mr} and that the provided torsion point images are indeed the images under this isogeny. For full details about the zero-knowledge proof, we refer to [BKW20, Sect. 5].

Subsequently, the server computes their secret isogeny $\phi_k : E_{mr} \to E_{mrk}$, where $E_{mrk} := E_{mr}/\langle \phi_r \circ \phi_m(P_K) + [k]\phi_r \circ \phi_m(Q_K)\rangle$. Moreover, the server computes the images of the order N_{V} torsion points and the basis of $E_{mr}[N_{\mathsf{R}}]$ provided by the client. The

server sends E_{mrk} together with the torsion point information to the client. Using an interactive zero-knowledge proof with a cut-and-choose approach between server and client, the server can prove to the client that it computed the isogeny and the torsion point images correctly. This proof uses the torsion point images of order N_{V} and the server's initial commitment to the key k. Details about this zero-knowledge proof can be found in [BKW20, Sect. 6].

After executing the zero-knowledge proof with the server to convince itself of the correctness of the server's reply, the client uses the images of the $E_{mr}[N_{\mathsf{R}}]$ torsion to unblind E_{mrk}. The unblinding isogeny $\hat{\phi}'_r$ is the pushforward of the dual of ϕ_r along ϕ_k to E_{mrk}. This allows the client to compute a curve isomorphic to $E_{mk} := E_m/\langle \phi_m(P_K) + [k]\phi_m(Q_K)\rangle$ without knowing k at any point in time. Hashing the input together with the j-invariant of E_{mk} and the server's initial commitment to their key k yields the output of the VOPRF. The isogeny evaluations of the OPRF are sketched in Fig. 4.7.

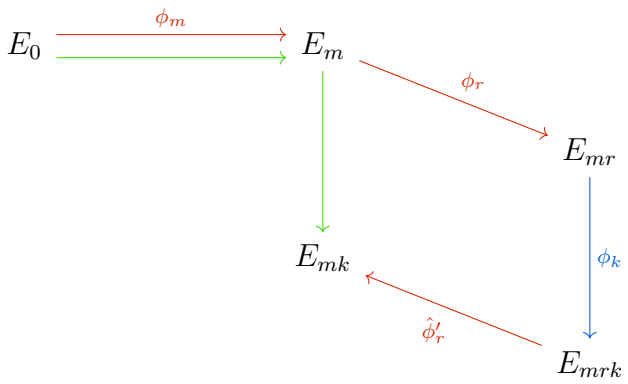

Fig. 4.7 Sketch of isogeny-based VOPRF by Boneh, Kogan and Woo. The isogenies computed by the client are marked in red (ϕ_m, ϕ_r, and $\hat{\phi}'_r$) while the server's isogeny is noted in blue (ϕ_k). The green isogenies represent the map which is jointly evaluated by the client and the server.

4.3.4 The auxiliary one-more SIDH assumption

To prove the unpredictability of their SIDH-based VOPRF, Boneh, Kogan and Woo introduce a new hardness assumption, the auxiliary one-more SIDH assumption in [BKW20]. In this section, we recall the game underlying this new security assumption, before providing an adversary that can win the security game later.

While decision queries defined in the following game are used in the OPRF's security proofs, our attacks will not make use of decision queries and a reader may choose to ignore this additional ability of the adversary.

Game 4.3.4 (Auxiliary One-More SIDH). *Let $p = f \cdot N_1 \cdots N_n - 1$ be a prime depending on the security level λ and n, where N_i are smooth coprime integers and f is a small cofactor, and let $\mathsf{M}, \mathsf{K} \in \{1, \ldots, n\}$ be two distinct indices. Consider the following game between a challenger and an adversary:*

- *The challenger chooses a random supersingular curve $E_0/\mathbb{F}_{p^2}$ and a basis $\{P, Q\}$ of $E_0[(p+1)/(N_\mathsf{M} \cdot N_\mathsf{K})]$. Moreover, it chooses $K \in E_0$ of order N_K, computes $\phi_K : E_0 \to E_K := E_0/\langle K \rangle$, and sends E_0, P, Q, and E_K to the adversary.*
- *The adversary can make a sequence of the following queries to the challenger:*
  - *Challenge query: The challenger chooses $M \in E_0[N_\mathsf{M}]$ randomly and sends it to the adversary.*
  - *Solve query: The adversary submits $V \in E_0[(p+1)/N_\mathsf{K}]$ to the challenger, who computes $\phi_{KV} : E_0 \to E_0/\langle K, V \rangle$ and sends $j(E_0/\langle K, V \rangle)$, $\phi_{KV}(P)$, and $\phi_{KV}(Q)$ to the adversary.*
  - *Decision query: The adversary submits a pair (i, j) to the challenger, where i is a positive integer bounded by the number of challenge queries made so far, and $j \in \mathbb{F}_{p^2}$. The challenger responds* true *if $j = j(E_0/\langle K, M \rangle)$, where M is the challenger's response to the ith challenge query, and* false *otherwise.*
- *The adversary outputs a list of distinct pairs of the form (i, j), where i is a positive integer bounded by the number of challenge queries made and $j \in \mathbb{F}_{p^2}$.*

We call an output-pair (i, j) correct, if j is the j-invariant of $E_0/\langle K, M \rangle$, where M is the challenger's response to the i-th challenge query. An adversary wins the game if the number of correct pairs exceeds the number of solve queries.

Assumption 4.3.5 (Auxiliary One-More SIDH [BKW20]). *For every constant n and every distinct $\mathsf{M}, \mathsf{K} \in \{1, \ldots, n\}$, every efficient adversary wins the above game with probability negligible in λ.*

In the following, we show that the auxiliary one-more SIDH assumption by Boneh, Kogan and Woo does not hold.

4.3.5 Attacks on the auxiliary one-more SIDH assumption

We give different attacks on the security problem underlying Assumption 4.3.5 that follow a similar strategy. Let K be the server's secret subgroup, determining the isogeny $\phi_K : E_0 \to E_0/\langle K \rangle$. The idea is to use a number of solve queries to subsequently predict $E_0/\langle K, M \rangle$ for any $M \in E_0[N_\mathsf{M}]$. To this end, we will derive a method to extract the subgroup generated by $\phi_K(P)$ for any $P \in E_0[N_\mathsf{M}]$ with a number of solve queries, i.e. an attacker recovers certain torsion point images up to a scalar under the secret isogeny. Using this procedure, an adversary can extract the subgroups generated by $\phi_K(P_M)$, $\phi_K(Q_M)$ and $\phi_K(P_M + Q_M)$, where $\{P_M, Q_M\}$ is a basis of $E_0[N_\mathsf{M}]$.

Knowing these subgroups allows the adversary to compute the subgroups generated by $\phi_K(M)$ for arbitrary $M \in E_0[N_\mathsf{M}]$ without any further solve queries. Given a generator of $\langle \phi_K(M) \rangle$, the adversary can compute the j-invariant of $E_0/\langle K, M \rangle$ as $E_0/\langle K, M \rangle \cong E_K/\langle \phi_K(M) \rangle$. In particular, the adversary can produce arbitrarily many correct output-pairs and win the security game underlying the auxiliary one-more SIDH assumption (Assumption 4.3.5). Note that our attack does not recover the server's secret, but rather enough information to make the server's input to the OPRF obsolete.

First, we show that recovering said torsion point images up to a scalar is sufficient to compute the correct answer to arbitrary challenges in the corresponding security game. Subsequently, we give multiple approaches to recover these torsion point images. In Section 4.3.7, we will show how the attack on the security assumption translates to an attack on the (V)OPRF itself.

Winning the security game given torsion point images

In this subsection, we show how mapping three different subgroups of order N_M to $E_K := E_0/\langle K \rangle$ is enough to recover sufficient information to compute a generator of the subgroup $\langle \phi_K(M) \rangle \subset E_K$ for any point $M \in E_0[N_\mathsf{M}]$.

Lemma 4.3.6. *Let P_V, Q_V, $R_V := P_V + Q_V \in E_0$ be pairwise linearly independent points of smooth order N_M and let $\phi_K : E_0 \to E_K$ be an unknown isogeny of degree coprime to N_M. Given the points P_V, Q_V, R_V and the subgroups $\langle \phi_K(P_V) \rangle$, $\langle \phi_K(Q_V) \rangle$ and $\langle \phi_K(R_V) \rangle$, an adversary can compute $\langle \phi_K(M) \rangle$ for arbitrary $M \in E_0[N_\mathsf{M}]$.*

Proof. Fix P', Q', and R' to be generators of $\langle \phi_K(P_V) \rangle$, $\langle \phi_K(Q_V) \rangle$ and $\langle \phi_K(R_V) \rangle$, respectively. Note that the given information $\langle \phi_K(P_V) \rangle$, $\langle \phi_K(Q_V) \rangle$ and $\langle \phi_K(R_V) \rangle$ is the same as knowing $\phi_K(P_V)$, $\phi_K(Q_V)$, $\phi_K(R_V)$ up to a scalar multiple coprime to N_M. There are many different generators for the groups $\langle \phi_K(P_V) \rangle$, $\langle \phi_K(Q_V) \rangle$ and $\langle \phi_K(R_V) \rangle$ but for any

fixed choice we have

$$\begin{aligned} P' &= \alpha\phi_K(P_V), \\ Q' &= \beta\phi_K(Q_V), \\ R' &= \gamma\phi_K(R_V) \end{aligned}$$

for some (unknown) integers α, β, γ coprime to N_{M}. As isogenies are homomorphisms, we have $\phi_K(R_V) = \phi_K(P_V) + \phi_K(Q_V)$. One finds a, b such that $R' = aP' + bQ'$, which can be done efficiently as computing discrete logarithms is easy in a group of smooth order N_{M}. We have $\gamma = a\alpha = b\beta$. Thus, it is possible for the attacker to recover the ratio $\alpha/\beta = b/a$.

Given $M \in E_0[N_{\mathsf{M}}]$, an adversary can compute the integers k_1, k_2 such that $M = k_1P_V + k_2Q_V$ (which again is possible because N_{M} is smooth) and obtain $\langle\phi_K(M)\rangle$ by computing $\langle k_1\phi_K(P) + k_2\phi_K(Q)\rangle = \langle k_1P' + k_2\frac{\alpha}{\beta}Q'\rangle$. □

In particular, an adversary who knows $\phi_K(P_V)$, $\phi_K(Q_V)$ and $\phi_K(R_V)$ up to an unknown scalar and $E_K := E_0/\langle K\rangle$ can compute $E_0/\langle K, M\rangle \cong E_K/\langle\phi_K(M)\rangle$ for any $M \in E_0[N_{\mathsf{M}}]$.

Recovering points in $\phi_K(E_0[N_{\mathsf{M}}])$ up to a scalar

The previous subsection shows that $E_0/\langle K, M\rangle$ can be computed by an adversary for arbitrary $M \in E_0[N_{\mathsf{M}}]$ as long as they can recover images of points in $E_0[N_{\mathsf{M}}]$ under the secret isogeny ϕ_K up to scalar. In this section, we will present multiple ways an adversary can recover this information. For better understanding, we include in our exposition not only a polynomial and a subexponential attack (in case countermeasures to prevent the former one are put in place) but also an exponential attack.

Query points of arbitrary order. Let $M \in E_0[N_{\mathsf{M}}]$. We are interested in recovering $\phi_K(M)$ up to a scalar, given access to the oracle provided by the solve queries in Game 4.3.4. Note that our attack will not use decision queries as defined in the same game.

There is a simple procedure to compute an isogeny between the curves E_K and $E_M := E_K/\langle\phi_K(M)\rangle$ and therefore $\phi_K(M)$ up to scalar, if solve queries are allowed for points of arbitrary order. Recall that during a solve query in Game 4.3.4, an adversary gets to submit points $V \in E_0[(p+1)/N_{\mathsf{K}}]$ to the challenger, who replies with the j-invariant of $E_0/\langle K, V\rangle$ and some additional torsion point images. Algorithm 4.1

describes how an adversary can recover $\phi_K(M)$ up to a scalar for arbitrary $M \in E_0[N_\mathsf{M}]$. The algorithm recovers the isogeny from E_K to $E_K/\langle\phi_K(M)\rangle$ by using solve queries to obtain all intermediate curves along this isogeny. This way, the adversary recovers the isogeny $E_K \to E_K/\langle\phi_K(M)\rangle$ one step at a time and therefore its kernel $\langle\phi_K(M)\rangle$.

Algorithm 4.1: Computation of $\langle\phi_K(M)\rangle$ using solve queries on points of arbitrary order

Let $\{l_i\}_{i=0}^n$ be an integer sequence of all divisors of N_M such that l_{i+1}/l_i is a prime, $l_i < l_{i+1}$, with $l_0 := 1$, $l_n := N_\mathsf{M}$.
Input: E_K, $M \in E_0[N_\mathsf{M}]$ and access to an oracle answering solve queries as defined in Game 4.3.4.
Output: A generator of $\langle\phi_K(M)\rangle$.
$E^{(n)} \leftarrow E_0/\langle K\rangle$
for $i = n-1, \ldots, 0$ **do**
  Query the oracle with the point $V_i := [l_i]M$ and obtain the curve $E^{(i)} := E_0/\langle K, V_i\rangle = E_0/\langle K, [l_i]M\rangle = E_K/\langle[l_i]\phi_K(M)\rangle$.
  Find l_{i+1}/l_i-isogeny ϕ_i from $E^{(i+1)}$ to $E^{(i)}$.
return A generator of $\ker(\phi_0 \circ \cdots \circ \phi_{n-1})$.

Lemma 4.3.7. *Algorithm 4.1 returns $\lambda\phi_K(M)$, where $\lambda \in \mathbb{Z}$ is coprime to N_M.*

Proof. Let ψ_M be the isogeny from E_K to $E_K/\phi_K(M)$. Then the claim follows from the observation that $E_0/\langle K, [l_i]M\rangle \cong E_0/\langle[l_i]K, [l_i]M\rangle$, since l_i is coprime to the order of K. □

Remark 4.3.8. Note that an attacker can easily change the attack to require fewer queries. Instead of using one query for each intermediate curve, an attacker can choose any factorisation $f_1 \cdots f_t$ of N_M such that f_i are roughly of equal size and query the oracle with $\left[\prod_{j=1}^b f_i\right] M$ for $b = 1, \ldots, t$. Then, the attacker is left to recover the isogeny between any two consecutive queries, i.e. the isogenies of degree f_i for $i = 1, \ldots, t$, using a meet-in-the-middle approach.

Game 4.3.4 does not restrict the points of $E_0[(p+1)/N_\mathsf{K}]$ that can be submitted to the solve queries. However, in the context of the game, this attack can be easily thwarted by answering a solve query only if the submitted point is of order $(p+1)/N_\mathsf{K}$. This property can be checked efficiently by the challenger. In Section 4.3.7, we discuss how this polynomial-time attack and its countermeasures translate to the VOPRF protocol.

Query points of order $(p+1)/N_{\mathsf{K}}$. Next, we present how an attacker can retrieve the necessary information even if they are only allowed to send solve queries on points of order $(p+1)/N_{\mathsf{K}}$, i.e. if the challenger checks the order of a submitted point and only replies to a query if the point is of order $(p+1)/N_{\mathsf{K}}$.

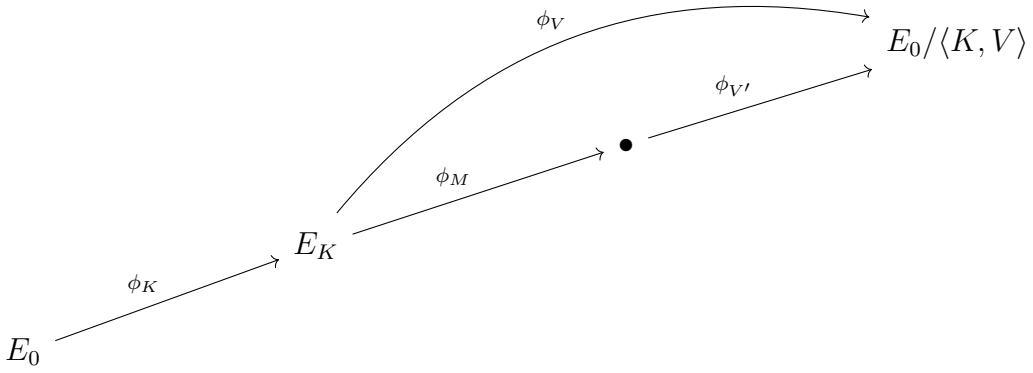

Fig. 4.8 Depiction of the isogenies of a solve query

Let ϕ_V denote the isogeny $E_K \to E_0/\langle V, K\rangle$ of degree $(p+1)/N_{\mathsf{K}}$ and let $\phi_V = \phi_{V'} \circ \phi_M$ be its decomposition into a degree $(p+1)/(N_{\mathsf{K}}N_{\mathsf{M}})$ and a degree N_{M} isogeny. Our attack aims to recover the image of multiple subgroups of $E_0[N_{\mathsf{M}}]$ under the isogeny ϕ_K, i.e. we are interested in the kernel of the isogeny ϕ_M for different points V. The isogenies are depicted in Fig. 4.8.

Recovering $\phi_{V'}$ from torsion point information. Let $P, Q \in E_0[(p+1)/N_{\mathsf{M}}N_{\mathsf{K}}]$ be the torsion point basis provided by the challenger and let $V \in E_0[(p+1)/N_{\mathsf{K}}]$ be linearly independent from P or Q. Then, we can use the torsion point images provided during solve queries to compute $\hat{\phi_V'}$ as follows.

Let $P' := \phi_V \circ \phi_K(P)$, $Q' := \phi_V \circ \phi_K(Q)$ be the torsion point images provided by the challenger. The adversary can compute $\hat{\phi}_{V'}$ as the isogeny from $E_0/\langle K, V\rangle$ with kernel $\langle P', Q'\rangle$. Note that $\langle P', Q'\rangle \subset \ker(\hat{\phi}_{V'})$, because $\hat{\phi}_{V'} \circ \phi_{V'} = [(p+1)/N_{\mathsf{M}}N_{\mathsf{K}}]$ is the order of the points P, Q. As V is linearly independent to at least one of P and Q, the other inclusion follows from $\langle P', Q'\rangle$ spanning a subgroup of size $(p+1)/N_{\mathsf{M}}N_{\mathsf{K}}$.

Choosing P_V, Q_V as a basis of $E_0[(p+1)/N_{\mathsf{K}}]$ such that $[N_{\mathsf{M}}]P_V = P + [(p+1)/N_{\mathsf{M}}N_{\mathsf{K}}]Q$ and $[N_{\mathsf{M}}]Q_V = [(p+1)/N_{\mathsf{M}}N_{\mathsf{K}}]P + Q$, every point of the form $P_V + [i]Q_V$ or $[i]P_V + Q_V$ will be linearly independent of P or Q.

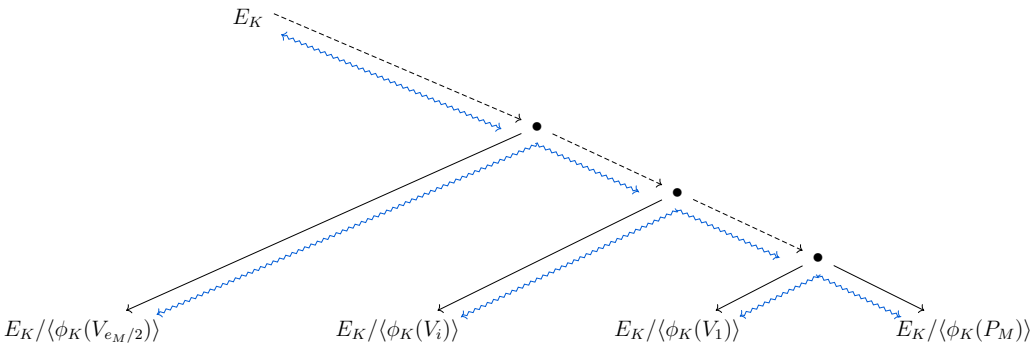

Fig. 4.9 Naïve attack where isogenies of increasing length need to be recovered. The blue lines represent the meet-in-the-middle computations.

As a consequence of $\phi_{V'}$ being easy to recover, we may assume that during a solve query an attacker can send a point M of order N_{M} to the challenger who returns $E_0/\langle K, M\rangle$. We are left to recover the kernel of ϕ_M.

Naïve attack to recover ϕ_M. Next we describe an exponential attack that recovers $\hat{\phi}_M$ using meet-in-the-middle (MITM) computations of increasing size. In the subsequent part, we will introduce a trade-off between queries and computation costs that reduces the complexity of the attack to subexponential.

Let P_M, Q_M denote a basis of $E_0[N_{\mathsf{M}}]$. For simplicity of exposition we treat N_{M} as a prime power and we write $N_{\mathsf{M}} = \ell_M^{e_M}$. The attack recovers $\phi_M : E_K \to E_K/\langle P_M\rangle$ by recovering each of the e_M intermediate curves one at a time.

The attacker starts by querying the solve oracle with two points $V_0 := P_M$ and $V_1 := P_M + [\ell_M^{e_M-1}]Q_M$. Note that the curves $E_K/\langle\phi_K(V_0)\rangle$ and $E_K/\langle\phi_K(V_1)\rangle$ are ℓ_M^2-isogenous, since they are both ℓ_M-isogenous to $E_K/\langle[\ell_M]\phi_K(V_0)\rangle = E_K/\langle[\ell_M]\phi_K(V_1)\rangle$. The attacker recovers the curve $E_K/\langle[\ell_M]\phi_K(V_0)\rangle$, which is the first intermediate curve on the ϕ_M isogeny path by computing the common neighbour of $E_K/\langle\phi_K(V_0)\rangle$ and $E_K/\langle\phi_K(V_1)\rangle$.

The rest of the attacks proceeds similarly. The attacker queries the solve oracle on the points $V_i := P_M + [\ell_M^{e_M-i}]Q_M$, for $i = 1, \ldots, e_M/2$ and runs a MITM attack to recover $E_K/\langle[\ell_M^i]\phi_K(V_0)\rangle$ given $E_K/\langle\phi_K(V_i)\rangle$ and $E_K/\langle[\ell_M^{i-1}]\phi_K(V_0)]\rangle$. This could be repeated e_M times to recover the entire isogeny ϕ_M. However, the attacker does not need to recover the last part of the isogeny through this strategy, since it is faster to directly compute the MITM between $E_K/\langle[\ell_M^{e_M/2}]V_0\rangle$ and the starting curve E_K. The attack with the required meet-in-the-middle computations is shown in Fig. 4.9.

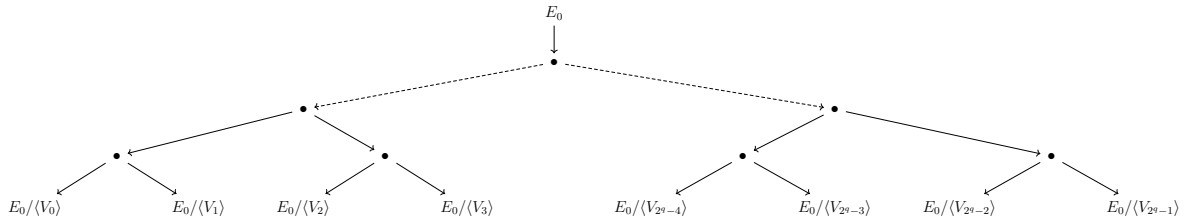

Fig. 4.10 The attacker queries the challenger on points corresponding to isogeny kernels leading to the leaves of this binary tree

Note that the isogenies that need to be recovered using MITM grow at each step. To recover the i-th intermediate curve, the attacker needs to compute an isogeny between two curves that are $\ell_M^{(i+1)}$-isogenous, which takes roughly $O(\ell_M^{(i+1)/2})$.

Clearly, this attack can be optimised by recovering multiple steps of ϕ_M at a time, and by making sure that the different MITM attacks that need to be executed have similar complexity. We will discuss these improvements next.

Full attack with query-time trade-off

We can reduce the complexity of the naïve attack by introducing a trade-off between queries and the cost of MITM computations. This is because the attacker recovers the whole path between two isogenies during a MITM computation. Thus, it is possible to recover more than one intermediate curve with a single (longer) MITM computation. Moreover, the queries can be spaced out more in order to reduce the length of the isogenies that have to be recovered using MITM strategies.

More formally, let 2^q denote the number of queries that an attacker can (or wants to) send to the challenger. For simplicity of this exposition, assume that $2e_m$ is divisible by $q+2$. The attacker chooses the V_i such that $E_0/\langle K, V_i\rangle$ correspond to curves that are the leaves of a binary isogeny tree. The V_i should be chosen such that there is an $\ell_M^{2e_M/(q+2)}$ isogeny between any two siblings in the binary tree and the curve that is $\ell_M^{e_M/(q+2)}$-isogenous to both leaves is their parent in the tree. Again, the parent and its sibling should be $\ell_M^{2e_M/(q+2)}$-isogenous, etc.

Remark 4.3.9. Note that it is easy to choose such a set of points V_i. Let P_M, Q_M be a basis of $E_0[\ell_M^{e_M}]$. The attacker can choose

$$\begin{aligned} V_0 &:= P_M \\ V_i &:= V_{i-2^{\lfloor \log i \rfloor}} + [\ell_M^{e_M-(\lfloor \log i \rfloor+1)2e_M/(q+2)}]Q_M \end{aligned}$$

Lemma 4.3.10. *Let $E_0/\langle V_i\rangle$ and $E_0/\langle V_j\rangle$ be ℓ_M^k isogenous curves. Then $E_K/\langle\phi_K(V_i)\rangle$ and $E_K/\langle\phi_K(V_j)\rangle$ are ℓ_M^k-isogenous curves too.*

Proof. This follows from $N_\mathsf{K} = \deg(\phi_K)$ being coprime to ℓ_M^k. □

In particular, $\{\phi_K(P_M), \phi_K(Q_M)\}$ is a basis of $E_K[N_\mathsf{M}]$ and $E_K/\langle\phi_K(V_i)\rangle$ are the leaves in a binary tree where all siblings are $\ell_M^{2e_M/(q+2)}$ isogenous.

After querying the oracle to obtain $E_K/\langle\phi_K(V_i)\rangle = E_0/\langle K, V_i\rangle$, an attacker iteratively recovers parent nodes in the binary tree using a meet-in-the-middle approach. Any siblings in the tree correspond to curves that are $\ell_M^{2e_M/(q+2)}$-isogenous, thus this can be done in $O(\ell_M^{e_M/(q+2)})$ time and memory. Note that the root of the binary tree is recovered after $2^q - 1$ such meet-in-the-middle instances, i.e. the number of internal nodes in the binary tree. This root of the binary tree is by construction $\ell_M^{2e_M/(q+2)}$-isogenous to E_0. This isogeny can be recovered using one final meet-in-the-middle search. An attacker recovers and saves the intermediate nodes and isogenies from E_K to the leaf $E_K/\phi_K(V_0)$. Clearly, the kernel of this isogeny is $\phi_K(V_0)$.

In summary, we can recover the isogeny from $E_K \to E_K/\langle\phi_K(P_M)\rangle$ for any P_M with 2^q queries to the challenger and 2^q instances of meet-in-the-middle isogeny computations with the cost of $O\left(\ell_M^{e_M/(q+2)}\right)$ time and memory each.

Remark 4.3.11. If $\ell_M = 2$, we get q bits for free, i.e. one additional bit per layer of the binary tree. This is because every parent node in the binary tree has three outgoing edges: two edges leading to its children and one edge leading towards the root. Thus, having recovered both paths to the children an attacker gets one step towards the root for free.

4.3.6 Analysis of the attack

The proposed attack is composed of two stages: firstly the generators of $\langle\phi_K(P_V)\rangle$, $\langle\phi_K(Q_V)\rangle$, and $\langle\phi_K(R_V)\rangle$ for three pairwise linearly independent points P_V, Q_V, R_V are recovered, and then these points are used to recover $\phi_K(M)$ for any possibly challenge $M \in E_0[N_\mathsf{M}]$.

The second part consists mostly of pairing evaluations and discrete log computations in groups of smooth order. Thus, it runs in polynomial time. The complexity of the attack is dominated by the complexity of recovering the subgroups during the first step.

The algorithm proposed to solve this first step in the presence of countermeasures against our polynomial time attack, as presented in the previous section, offers different trade-offs between computation costs and solve queries. As little as two solve queries

can be enough to recover ϕ_M with two meet-in-the-middle computations. If we write $N_{\mathsf{M}} \approx 2^m$, each meet-in-the-middle requires $O(2^{m/3})$ operations. This is already an improvement over the standard meet-in-the-middle attack that requires $O(2^{m/2})$ time. The OPRF protocol targets 128 bits of security, which corresponds to $m \approx 5\lambda/2 = 320$. Thus six queries (two per generator) are enough to reduce the security to $m/3 = 106$ bits. The number of solve queries can be significantly increased to obtain a faster attack. Note that OPRF protocols are usually used for applications such as private set intersection that support a large number of queries. Thus, common scenarios where the OPRF may be used would easily enable an attack with many queries.

Since OPRFs are used in protocols where the clients interact several times with the server, we can expect the attacker to be able to run several OPRF instances. Thus, we model a solve query as an oracle query, where it has a constant complexity. Then, the overall complexity of recovering a generator of $\langle \phi_K(P_V) \rangle$ with 2^q solve queries is $O(2^{m/(q+2)+q})$ operations, since the attacker needs to compute 2^q meet-in-the-middle instances between curves which are $2^{2m/(q+2)}$-isogenous. In terms of the security parameter, that complexity is equivalent to $O(2^{5\lambda/2(q+2)\ +\ q})$, since the OPRF protocol suggests using $m \approx 5\lambda/2$. If the number of solve queries is unrestricted, the complexity of the attack is minimised for $q = \sqrt{5\lambda/2} - 2$, which gives an overall complexity of $O(2^{\sqrt{10\lambda}-2})$, or using the L-notation $L[1/2, c]$, for some constant c. This shows the attack is subexponential, assuming that the solve query complexity is $O(1)$.

At 128-bit of security, our attack becomes feasible with around 64 solve queries, when it requires 64 meet-in-the-middle computations between curves which are 2^{80}-isogenous, i.e. each MITM has a complexity of 2^{40} operations. If the number of solve queries is unrestricted, an attacker can use for instance 2^{18} solve queries to reduce the overall complexity of the attack to 2^{18} MITM computations, each with a complexity of roughly 2^{16} operations.

The high-level attack does not generally require much memory. Storing the isogeny tree in memory is not particularly demanding, especially if the tree is traversed in a depth-first manner. In particular, memory is used only to store the part of the recovered isogeny, together with the two curves between which the meet-in-the-middle needs to be computed. However, a more significant amount of memory is used by the meet-in-the-middle computations, and indeed we see that the memory used by a single meet-in-the-middle generally outweighs the memory used by the rest of the attack. Meet-in-the-middle computations between curves which are 2^n-isogenous require to store $2^{n/2}$ curves. Thus, their memory requirements are given by $2 \cdot 2^{n/2} \log p$, since each curve can be represented by its j-invariant in $\mathbb{F}_{p^2}$. For common security levels, such as

those proposed by Boneh, Kogan and Woo [BKW20], the memory requirements remain moderate. In Section 4.3.8, we show that indeed our attack requires about 3 GB of memory to break 128 bits of security. However, for a more complete asymptotical analysis, we note that the memory requirements may become a bottleneck for the attack against higher security levels. In those instances, it may be preferable to substitute the meet-in-the-middle approach with the van Oorschot-Wiener algorithm [vW99]. This reduces the memory consumption at the cost of higher asymptotic complexity. In particular, the vOW algorithm requires $O(2^{3n/4})$ computations (compared to $O(2^{n/2})$ of MITM) to recover the halfway curve between curves which are 2^n-isogenous. Thus, while the concrete performance of the attack may differ, its asymptotic complexity remains subexponential.

4.3.7 Attack on the SIDH-based OPRF

Having presented an attack on one of the security assumptions underlying the isogeny-based OPRF by Boneh, Kogan and Woo, we investigate how an adversary can use the same method to attack the OPRF itself.

We will show that a malicious client can send carefully crafted queries to the server for which it can produce all necessary NIZK proofs required by the protocol that was summarised in Section 4.3.3. However, after some offline computation analogously to the attack on the auxiliary one-more SIDH assumption outlined in the previous section, the malicious client can evaluate the OPRF on any input without the help of the server. Even though the malicious client does not recover the server's secret key k, this breaks the pseudorandomness of the OPRF, see Definition 4.3.1. We will use the notation introduced in Section 4.3.3 to refer to the different isogenies of the OPRF.

A malicious client will not use a hashed input to obtain the kernel for the first isogeny $\phi_m : E_0 \to E_m$ but rather choose the kernel of this first isogeny. The choice is analogous to the points from $E_0[N_{\mathsf{M}}]$ that the attacker submitted to the solve queries in the attack of the previous section. In other words, instead of computing E_m as $E_0/\langle P + [H(x)]Q\rangle$ for some input x, the malicious client chooses a point V_i and computes E_m as $E_0/\langle V_i\rangle$ in the i-th evaluation of the OPRF.

The rest of the protocol is executed honestly. The malicious client can pick some $r \in N_{\mathsf{R}}$ to blind their maliciously chosen E_m. And it can compute the torsion point information for the server honestly since it knows the kernel of the isogeny from E_0 to $E_{mr} = E_m/\langle\phi_m(P_R) + [r]\phi_m(Q_R)\rangle$. In particular, the malicious client will always be able to produce the valid non-interactive zero-knowledge proof of knowledge for the kernel of $E_0 \to E_{mr}$ and the correct computation of the torsion point information.

Following through with the rest of the OPRF protocol, the malicious client obtains the j-invariant of the curve $E_0/\langle V_i, K\rangle$ after unblinding. Here, K denotes the server's secret $P_K + [k]Q_K$ again. This is exactly what corresponds to a solve query in the auxiliary one-more SIDH game, Game 4.3.4.

Now the malicious client can proceed as in the attacks on the auxiliary one-more SIDH assumption.

In the attack using points of arbitrary order dividing N_{M}, the malicious client recovers the isogeny $E_K \to E_K/\langle\phi_K(P)\rangle = E_0/\langle K, P\rangle$ and therefore $\langle\phi_K(P)\rangle$ for any $P \in E_0[N_{\mathsf{M}}]$ in polynomial time. This is done by submitting points of lower order, i.e. choosing the isogeny $E_0 \to E_m$ shorter, to recover the isogeny stepwise. We have shown in a previous section that after recovering three such isogenies corresponding to pairwise linearly independent points $P, Q, P+Q$ a malicious client can compute $E_0/\langle M, K\rangle$ for any $M \in E_0[N_{\mathsf{M}}]$.

Then, the malicious client can evaluate the OPRF on arbitrary inputs x as follows: They compute the point $M := P_M + [H_1(x)]Q_M$ as in the honest evaluation and then they compute $j(E_0/\langle M, K\rangle)$ directly as described in our attack. Hashing this j-invariant together with the input x and public information of the server yields the output of the OPRF. Note that the malicious client does not even need to interact anymore with the server to evaluate the OPRF on arbitrary inputs.

Clearly, this breaks the pseudorandomness property of an OPRF, see Definition 4.3.1, as a malicious client will be able to predict the output of the OPRF for any input after the initial queries.

Remark 4.3.12. The SIDH-based OPRF protocol by Boneh, Kogan and Woo does not prohibit malicious clients from using points of smaller order dividing N_{M}, i.e. from using a shorter isogeny $E_0 \to E_m$. However, this attack could be thwarted if the server checked that the submitted curve is of correct distance from the starting curve. A simple test using pairing computations on the provided torsion point information may be tricked by an adversary, but the NIZK proof of the client could be extended to include a proof that the client's witness, i.e. the kernel of the isogeny $E_0 \to E_{mr}$, is of full order $N_{\mathsf{M}}N_{\mathsf{R}}$.

Even if countermeasures for this polynomial-time attack are put in place, we are left with the following subexponential attack when points of full order are used.

The client evaluates the OPRF on a certain number of inputs that correspond to solve queries in the auxiliary one-more game. More precisely, the client chooses the kernel of their first isogeny as in the subexponential attack of the previous section. After blinding, evaluation of the server and unblinding, the client obtains what would have been the result of a solve query in the previous section. After the offline computation

which, using meet-in-the-middle routines, recovers the binary tree as described in the attack with query-time trade-off, the client obtains torsion point images of $E_0[N_\mathsf{M}]$ up to scalar under the isogeny $E_0 \to E_K := E_0/\langle P_K + [k]Q_K \rangle$. As before, this is enough to compute $E_0/\langle M, K \rangle$ for any $M \in E_0[N_\mathsf{M}]$, allowing the client to compute the OPRF on arbitrary inputs and therefore breaking the pseudorandomness of the OPRF.

Possible countermeasures

In the case where the degree of the client's isogeny is forced to be $N_\mathsf{M} N_\mathsf{R}$, the proposed attack has subexponential complexity, and thus possible countermeasures may include increasing the parameter sizes. However, the solve queries to time trade-off makes this approach rather costly. If the number of possible solve queries is unrestricted, to get 128-bit security one would need the isogeny degree N_M to be $\approx 2^{(128^2)}$. This can be partially lowered by guaranteeing security only up to a certain number of queries. Given a limit of 2^Q queries, the exponent m needs to guarantee that $\min\{2^{\sqrt{m}-2}, 2^{m/(Q+2)+Q}\}$ is at least 2^λ. Thus, for 128-bit security, with $Q = 64$ the isogeny degree N_M would have to be increased to $\approx 2^{4224}$, whereas $Q = 32$ would require a degree $N_\mathsf{M} \approx 2^{3264}$. Note that handling 2^{32} queries may well be within the scope of several OPRF applications, and isogenies of the given size may become impractically large. Their feasibility, however, depends on the specifics of the OPRF application and its time and bandwidth requirements. Thus, while the attack is subexponential (assuming $O(1)$ complexity for solve queries), increasing the parameter size comes at a significant performance and communication cost.

Therefore, it is important to consider possible algorithmic countermeasures. Firstly, note that the attacker submits seemingly valid requests, so the server cannot stop such interactions. Even if the server did want to prevent these requests, it may not be able to detect them. This is because the attacker only submits the image curve and some torsion point images under an isogeny with chosen kernel.

However, the attack strongly depends on the attacker choosing the point V. If the input points V were randomised, the attack as such could not work. The OPRF protocol requires that such points are obtained via hashing the client's PRF input x, but it does not enforce it. Hence, a possible countermeasure to the proposed attack would be requiring the client to provide a zero-knowledge proof that the curve E_{mr} is not only the result of honest isogeny computations, but also that the kernel of ϕ_m is the result of some hash function. However, we are not aware of a way to prove such a statement efficiently.

4.3.8 Proof of concept implementation

We implemented our subexponential attack in SageMath to demonstrate the correctness of the algorithm and prove its feasibility.[1] This implementation is to be regarded only as a proof-of-concept version and several subroutines could be further optimised. Improving their performance and using lower-level languages, such as C, as well as platform-specific instructions, such as AVX, could significantly reduce the running time of the attack. Further, more recent versions of SageMath since publication of the this attack offer significantly faster isogeny evaluation.

The proposed attack has two distinguishing features that help its implementation: it can be easily parallelised, and it has very low memory requirements. Indeed, the computations to recover the generators of $\langle \phi_K(P_V) \rangle$, $\langle \phi_K(Q_V) \rangle$ and $\langle \phi_K(P_V + Q_V) \rangle$ are independent of each other. It is also possible to achieve a higher degree of parallelisation. Within each computation to recover a single generator, the meet-in-the-middle operations within each layer of the tree are also independent of each other, and they can thus be parallelised. In this case, the tree is generated layer-by-layer in a breadth-first manner. Note that while this may require a sizeable amount of memory to fully store an entire layer, the memory requirements are hardly the bottleneck. An attack with 2^{20} queries requires to store, at most, 2^{19} curves. Since an elliptic curve can be represented by its j-invariant, the memory limit is $2^{19} \cdot 2 \log p$. With a prime of size $\approx 2^{1500}$, as proposed in the OPRF protocol, the memory limit is about 196 MB. Alternatively, it is possible to traverse the tree in a depth-first manner to further lower the memory requirement, but this may limit the degree of parallelisation. Our implementation provides parallelised meet-in-the-middle computations with a configurable number of cores in parallel.

Results

The majority of the attack's subroutines have polynomial complexity and they are optimised enough that their performance does not affect the overall running time. The building block that most affects the performance of the attack is the meet-in-the-middle computation. Indeed, the timings of the attack are directly correlated to the timing of a single meet-in-the-middle computation and the total number of queries. The memory requirements of the attack are given by the amount of memory needed for a single meet-in-the-middle, which in turn depends on the distance between the two curves. For parallelised implementations of the attack, the memory requirements correspond to as many meet-in-the-middle computations as there are parallel instances.

[1] Source code available at https://github.com/isogenists/isogeny-OPRF.

Table 4.1 shows the running times at different security levels on an Apple M1 CPU clocked at 3.20 GHz with 4 CPUs running in parallel. Up to 32 bits of security, the results come from running the entire attack, whereas for higher security levels the results are estimated based on those of a single MITM computation. The estimated time t is computed as

$$t = \frac{3(M+Q)2^q}{C}, \tag{4.1}$$

where M is the average running time of a MITM computation, Q is the average running time of a solve query computation, 2^q is the number of queries and C is the number of CPUs running in parallel. This formula follows from the fact that there are 2^q MITM computations and 2^q solve queries for each generator recovery, and three of those are needed. Running computations at lower security levels and computing Eq. (4.1) does indeed estimate the running time accurately.

We estimate that our non-optimised implementation running on a laptop with 4 CPUs can break 64 bits[2] of security in less than two days and 128 bits of security in about 5 years. If the same attack was performed with more powerful hardware and an optimised implementation, the running time could easily be reduced.

Lastly, note that in the implementation solve queries are simulated locally. A real attack would interact with the server, and thus the attack time would not include the time to compute a solve query. For completeness, Table 4.1 reports the running time of the entire implementation, including the solve queries.

4.3.9 Trusted setup of the starting curve

Boneh, Kogan and Woo suggest using a random supersingular elliptic curve as the starting curve in their OPRF protocol. Unfortunately, there is currently no known algorithm to generate a random supersingular elliptic curve such that its endomorphism ring is unknown to the person who generated it. Some failed attempts to solve this problem have been studied in [LB20, CPV20, BBD+22]. However, a distributed trusted setup is often proposed as a workaround, and the tools necessary to practically run such a distributed trusted-setup ceremony were developed in [BCC+22].

This motivates the question whether a trusted third party or distributed trusted-setup is needed to generate the starting curve E_0.

[2]We report the results for $e_M = 169$, which corresponds to $\lambda = 67$. That is because our implementation requires $(q+2) \mid e_M$, and 169 allows choosing $q = 11$. The requirement that $(q+2) \mid e_M$ is a limitation of the implementation and not of the attack itself.

Parameters				**MITM**		**Running Time**
$\log p$	λ	e_M	q	Distance	Memory (kB)	(s)
112	8	20	3	8	3.5	15
216	16	40	6	10	13.8	212 (3.53 m)
413	32	80	8	16	211.4	1,371 (22.85 m)
859	67	169	11	26	14,073	163,869 (1.89 d)
1,614	128	320	18	40	3,384,803	174,709,440 (5.54 y)

Table 4.1 Results of our proof-of-concept implementation running on an Apple M1 CPU clocked at 3.20 GHz with 4 CPUs in parallel and SageMath version 9.2. Results for $\lambda = 128$ are estimated based on the average running time of a meet-in-the-middle computation. Parameters include the size of the prime p, the security level λ, the degree of the isogeny written as $N_{\mathsf{M}} = 2^{e_M}$, and the number of queries 2^q. The MITM section reports the distance between the curves and memory needed to compute a single meet-in-the-middle.

Phrased differently, would choosing the starting curve E_0 and therefore knowledge of its endomorphism ring allow a malicious server, client or third party to break security properties of the (V)OPRF?

We first discuss whether a server may know the endomorphism ring of the starting curve E_0. The security proof of the OPRF relies on the hardness of finding two distinct isogenies (up to isomorphism) of the same degree from E_0 to a second curve [BKW20, Lem. 29]. If the server chooses the starting curve and therefore knows its endomorphism ring, they are able to produce such collisions by breaking the collision resistance of the CGL hash function as described in [PL17, EHL+18]. To guarantee provable security, a server should therefore not choose the starting curve.

However, breaking the verifiability insured by the zero-knowledge proof [BKW20, Protocol 17] or the weak binding property [BKW20, Game 3] of the protocol seems harder than finding collisions. Indeed, the server would need to produce two isogenies of degree dividing N_{K} such that both isogenies have the same action on the N_{V}-torsion for a chosen starting curve. We leave adapting the security proofs or finding an attack on the zero-knowledge proof for future work.

We now argue that any other party, either the client or a third party, cannot choose the curve E_0 without compromising the security of the protocol. In [dQKL+21], the authors describe algorithms for finding a secret isogeny when torsion information is provided. Their algorithms can be split into two categories: one where the starting curve has j-invariant 1728 and one where the starting curve is a so-called *trapdoor curve*

from which one can solve the isogeny problem faster than generic meet-in-the-middle algorithms.

Definition 4.3.13. Let p be a prime and A, B be coprime smooth integers. A tuple (E, T) is called a (A, B) *trapdoor curve* if the trapdoor T encompassing required information to run torsion point attacks from [dQKL+21] allows one to solve any instance of the SSI-T problem with secret isogeny of degree A from E and torsion point images of $E[B]$ in polynomial time.

When $B \approx A^2$ or larger, then one can construct (A, B) trapdoor curves from which one can retrieve secret isogenies of degree A in polynomial time, if the action on the B-torsion is known [dQKL+21, Thm. 15].

Attacks from the special starting curve with j-invariant 1728 do not apply here, since the starting curve is required to have an endomorphism ring which is unknown to the server. However, trapdoor curves have the property that without extra information they are difficult to distinguish from a random supersingular curve.

Suppose that a malicious party generates the starting curve E_0 in the following way. They generate a curve E' which is a trapdoor $(N_\mathsf{K}, N_\mathsf{V}N_\mathsf{R})$-curve and then perform a random isogeny walk of degree $N_\mathsf{M}N_\mathsf{R}$ (of which they keep track) to obtain E_0 which is sent to the server. Now the malicious party poses as a client and instead of honestly complying with the protocol, they use E' as E_{mr}. They can prove knowledge of a suitable isogeny and torsion point images as they know an isogeny of the correct degree from E_0. Then the server computes E_{mrk} and reveals the action on the $N_\mathsf{V}N_\mathsf{R}$-torsion. Since E_{mr} was chosen to be a trapdoor curve and $N_\mathsf{V}N_\mathsf{R} \approx N_\mathsf{K}^2$, the malicious party can retrieve this isogeny in polynomial time.

Such an attack can be thwarted by applying a trusted setup in which E_0 is a random curve with unknown endomorphism ring. In [BCC+22], an efficient way to perform a distributed trusted setup is described, ensuring that, if at least one participant is honest, the setup can be trusted. In that case, torsion point attacks are not applicable. Another countermeasure would be to increase N_K substantially making the construction of $(N_\mathsf{K}, N_\mathsf{V}N_\mathsf{R})$-curves infeasible. However, this variant would be susceptible to potential future improvements of trapdoor curve constructions.

4.4 Conclusion

In this chapter, we investigated the hardness of some isogeny "one-more" assumptions that were used in the security proofs of undeniable signatures and oblivious pseudorandom functions.

First, we showed that the OMSSCDH and 1MSSCDH problems can be solved with non-negligible probability by a polynomial time attacker. Jao and Soukharev [JS14] proposed an undeniable isogeny-based signature scheme based on these assumptions. We presented an attack against the unforgeability and invisibility properties of the Jao-Soukharev protocol, showing that an adversary with access to a signing oracle is able to forge arbitrary signatures at lower cost than expected for a given security parameter, λ. To summarise, this is achieved by computing a near-collision on the public hash function H and guessing an ℓ_B^{2k}-isogeny between an honest signature produced by the oracle for one message to the target forgery curve. The classical cost for this attack is $2^{4\lambda/5}$, with the hash function length equal to 2λ. We postulate that the quantum cost for this attack is $2^{6\lambda/7}$. These attacks imply that parameters need to be increased by 25% to achieve the same classical security level. Furthermore, we argue that the equivalence drawn in [JS14] between unforgeability and the OMSSCDH problem is incorrect, and hence that the security proofs in that paper are incorrect. Yet, we note that the inclusion of a hash function increases the difficulty of forgery, assuming the hash function is cryptographically secure, as the adversary is forced to search for a message that will result in a specific curve, rather than querying the oracle indiscriminately.

Second, we performed a thorough cryptanalysis of the SIDH-based oblivious pseudorandom function by Boneh, Kogan and Woo and the auxiliary one-more assumption this OPRF's security proof relies on. We show how an adversary can win the corresponding security game in polynomial time, or in presence of some additional countermeasures in subexponential time. The attack on the underlying hardness assumption leads to an attack on the pseudorandomness of the OPRF itself. We show how a malicious client can extract enough information from a number of initial executions of the OPRF protocol to subsequently evaluate the OPRF on arbitrary inputs without further interaction with the server. This attack breaks the pseudorandomness of the OPRF for the security parameters proposed by Boneh, Kogan and Woo. Moreover, we discussed the security implications following from a lack of a trusted setup (which was not required in the original proposal) when generating the starting curve parameter in the SIDH-based OPRF. We show how a client or a third party generating the starting curve can backdoor it to retrieve the server's secret key, while a malicious server generating the starting curve could break another assumption made in the security proof of the OPRF.

Finally, we want to point out that the undeniable signature scheme by Jao and Soukharev and the SIDH-based OPRF by Boneh, Kogan and Woo are also shown to be insecure by recent SIDH attacks [CD22, MM22, Rob22a] which were developed

after the results of this chapter. However, these attacks break the protocols even more fundamentally as they allow various parties to recover each other's private keys.

Overall there have been very few proposals for post-quantum OPRFs. Given the large communication cost of the CSIDH-based OPRF by Boneh, Kogan and Woo (500kB per evaluation), and that the lattice-based OPRF by Albrecht, Davidson, Deo and Smart [ADDS21] is also much less efficient than its classical counterparts would be, it is still an open problem to construct an efficient post-quantum OPRF. In case building such a general purpose efficient (V)OPRF remains elusive, it would alternatively be interesting to instead propose post-quantum alternatives for concrete applications, e.g. for anonymous authentication schemes.

CHAPTER 5

On the Isogeny Problem with Torsion Point Information

In this chapter, we present a new reduction from the problem of computing an isogeny of a specific degree to the endomorphism ring computation problem, if certain torsion point information is provided. The work presented in this chapter is an amended version of what was published previously as

Tako Boris Fouotsa, Péter Kutas, Simon-Philipp Merz, and Yan Bo Ti. On the isogeny problem with torsion point information. In Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe, editors, *PKC 2022, Part I*, volume 13177 of *LNCS*, pages 142–161. Springer, Heidelberg, March 2022.

Compared to the published version we further show that the reduction still applies for SIDH-like schemes that mask their degree or torsion point images as has been suggested as countermeasures to thwart the recent attacks on SIDH.

5.1 Introduction

There are infinitely many isogenies $E_1 \to E_2$ between two given curves, but for attacking isogeny-based primitives such as SIDH one is required to recover an isogeny $\varphi : E_1 \to E_2$ of a particular degree. Generic algorithms that compute an isogeny from the endomorphism rings are unlikely to return an isogeny of the degree required.

In [GPST16, Sect. 4], Galbraith, Petit, Shani and Ti presented a polynomial-time reduction for the problem of finding the secret isogeny in SIDH to the problem of computing the endomorphism ring of a supersingular elliptic curve. Their method exploits the fact that secret isogenies in SIDH are of degree approximately $p^{1/2}$ which is relatively small compared to the diameter of the supersingular ℓ-isogeny graphs involved. In the case where the isogeny one wishes to recover is not of particularly small degree, as is the case for example in B-SIDH [Cos20], SÉTA [DDF+21] or instantiations of SIDH with secret isogenies of larger degree, the observation no longer holds and the reduction algorithm of [GPST16] no longer applies.

In this chapter, we present a more general reduction algorithm that generalises to all SIDH-type schemes. More precisely, assuming the generalised Riemann hypothesis (GRH), we provide a polynomial-time (in $\log p$) algorithm that recovers an isogeny with given N_2-torsion point images between two supersingular elliptic curves of a specific degree N_1, given their endomorphism rings. More formally, let d be the minimal degree of any isogeny between two isogenous supersingular elliptic curves E_1 and E_2. Then, our algorithm solves the following task efficiently, whenever $N_1 < dN_2/16$.

Task 5.1.1. *Let N_1, N_2 be coprime integers and let $\varphi : E_1 \to E_2$ be a secret isogeny of degree N_1 between two supersingular elliptic curves. Let P_B, Q_B be a basis of $E_1[N_2]$. Given* $\mathrm{End}(E_1)$, $\mathrm{End}(E_2)$, *$\varphi(P_B)$, and $\varphi(Q_B)$, find an isogeny $\varphi' : E_1 \to E_2$ of degree N_1 such that $\varphi_{|E_1[N_2]} = \varphi'_{|E_1[N_2]}$.*

Note that this can be seen as a reduction of the SSI-T problem (not just for SIDH parameters) to the problem of computing endomorphism rings of supersingular elliptic curves. Since SIDH-type schemes such as B-SIDH tend to use balanced parameters, that is where $N_1 \approx N_2$, the condition that $N_1 < dN_2/16$ is very mild.

The main idea behind our algorithm is the following. Isogenies from E_1 to E_2 form a $\mathbb{Z}$-module M of rank 4. A basis of M can be computed using an algorithm due to Kirschmer and Voight [KV10] (or the KLPT algorithm [KLPT14]). Then, one computes an LLL-reduced basis $\psi_1, \psi_2, \psi_3, \psi_4$ of M. We show how to evaluate $\psi_i(P_B)$, $\psi_i(Q_B)$ for $i = 1, \ldots, 4$ given $\varphi(P_B)$ and $\varphi(Q_B)$, where the ψ_i in general do not have a smooth degree.

Since $\varphi = x_1\psi_1 + x_2\psi_2 + x_3\psi_3 + x_4\psi_4$ for some $x_i \in \mathbb{Z}$, the provided torsion point images yield 4 linear equations in 4 variables, x_1, x_2, x_3, x_4, modulo N_2. This is because torsion point images can be represented by a 2×2 matrix with entries from $\mathbb{Z}/N_2\mathbb{Z}$ and each entry corresponds to one equation. We will show that this system of equations has a unique solution for x_i modulo N_2 which we can efficiently compute. Since the ψ_i form an LLL-reduced basis, we will see that the absolute value of the coefficients x_i can be bounded by $N_2/2$ for $N_1 < dN_2/16$. Finally, this can be lifted to a solution for $x_i \in \mathbb{Z}$ and thus yields the secret isogeny.

The reduction presented in this chapter can be seen as an extension of previous work by Kohel, Lauter, Petit and Tignol [KLPT14] and Wesolowski [Wes22b] which allows to compute an isogeny (of no specific degree) between two supersingular elliptic curves, whenever the endomorphism rings of the curves are known. Note that Kohel et al. provide a heuristic polynomial-time algorithm for this reduction, whereas Wesolowski shows that this reduction works in polynomial-time in general assuming GRH only.

A unique trait of SIDH is that it reveals auxiliary points, which are the images of certain torsion points under the secret isogeny. Since publication of the conference version of this work at PKC 2022 [FKMT22], a series of papers by Castryck and Decru [CD22], Maino and Martindale [MM22], and Robert [Rob22a] broke SIDH and various derivatives such as B-SIDH or SÉTA efficiently by using the auxiliary points published in the protocols. The methods described in this chapter also require these points for the reduction. Thus, at a first glance it might seem that the reduction presented in this chapter has lost its relevance. However, apart from leading to a better understanding of the relation between foundational problems in isogeny-based cryptography, it is easy to construct parameter sets for which our reduction applies but the recent SIDH attacks do not. Further, countermeasures to thwart the attacks by masking the degree and/or the auxiliary points have been proposed [Mor22, Fou22]. We show how our reduction extends to protocols deploying these countermeasures.

Computing endomorphism rings and then applying our reduction gives rise to an attack against SIDH-like schemes. This attack provides a lower bound on the size of the prime p of the underlying finite field. For B-SIDH this shows that the size of the prime p is tight and cannot be lowered significantly, while simultaneously maintaining the claimed security level. This was particularly relevant at the time the results were first published predating recent SIDH attacks [CD22, MM22, Rob22a], but note that by using a starting curve of unknown endomorphism ring and masking the torsion point images, B-SIDH is not (yet) considered broken by recent attacks.

The attack arising from our reduction has a similar classical runtime as a generic meet-in-the-middle algorithm but is essentially memory-free, whereas meet-in-the-middle requires an exponential amount of memory. The quantum version of our attack is dominated by the computation of endomorphism rings of supersingular elliptic curve and has a much better runtime than previously known quantum attacks ($O(p^{1/4})$ [BJS14] compared to $O(p^{1/2})$ [JS19]). Furthermore, our attack does not suffer from issues arising from quantum memory, which was argued to be a problem for Tani's claw finding algorithm [JS19].

Chapter outline. In Section 5.2, we recall some background on the KLPT and LLL algorithms as well as a description of the related reduction by Galbraith, Petit, Shani, and Ti [GPST16]. In Section 5.3, we give algorithms to evaluate non-smooth degree isogenies and to compute an isogeny of a specific degree between two supersingular elliptic curves, given their endomorphism rings and certain torsion point information. In Section 5.4, we will address the implications of the attacks that used auxiliary points to break SIDH. Finally, we discuss the impact of this chapter's content on isogeny-based cryptography in Section 5.5.

5.2 Preliminaries

In this section, we briefly recall some consequences of the algorithm by Kohel, Lauter, Petit and Tignol (KLPT) [KLPT14] and the LLL lattice reduction [LLL82]. Moreover, we sketch a related algorithm from [GPST16] which computes an isogeny of specific degree between two supersingular elliptic curves with known endomorphism rings, if the degree of the sought isogeny is sufficiently small.

5.2.1 Connecting ideals and the KLPT algorithm

Let $B_{p,\infty}$ be a quaternion algebra ramified at p and at infinity (Definition 2.2.22). Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be maximal orders in $B_{p,\infty}$. Then the *quaternion path problem* asks for a left ideal I connecting $\mathcal{O}_1$ and $\mathcal{O}_2$, i.e., a left ideal I of $\mathcal{O}_1$ which is also a right ideal of $\mathcal{O}_2$. This is the quaternion analogue of the pure isogeny problem under Deuring's correspondence described in Section 2.2.6. We have the following result.

Lemma 5.2.1. *[KLPT14, Lem. 8] Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be maximal orders in $B_{p,\infty}$. Then the intersection $\mathcal{O}_1 \cap \mathcal{O}_2$ has the same index M in $\mathcal{O}_1$ and $\mathcal{O}_2$. Furthermore,*

$$I(\mathcal{O}_1, \mathcal{O}_2) := \{\alpha \in B_{p,\infty} \,|\, \alpha\mathcal{O}_2\overline{\alpha} \subset M\mathcal{O}_1\}$$

is a left ideal of $\mathcal{O}_1$ and a right ideal of $\mathcal{O}_2$ of reduced norm M. $I(\mathcal{O}_1, \mathcal{O}_2)$ can be computed in polynomial time.

Lemma 5.2.1 shows that one can compute a connecting ideal between two maximal orders efficiently. However, this ideal will not have smooth norm in general. In [KLPT14], the main algorithm shows how to compute an equivalent left ideal (see Definition 2.2.33) of $\mathcal{O}_1$ of norm ℓ^k, where ℓ can be any small prime number.

KLPT Algorithm. We will not recall all technical details of the KLPT algorithm here and instead we refer to [KLPT14, DKL+20]. However, we briefly sketch the different steps of the algorithm. Let I be the given left-ideal of $\mathcal{O}_1$. By picking random elements $\alpha \in I$ until the norm of α is a prime N and then considering the ideal $I\overline{\alpha}/\operatorname{Norm}(I)$ which is equivalent to I and of $\operatorname{Norm}(\alpha)$, one may assume that I is an ideal of prime norm N. Next, one picks $\beta \in I$ at random such that I is generated by N and β in $\mathcal{O}_1$. Both steps are easy and can be done very efficiently. Next, one factors $\beta = \gamma\delta \bmod N$ for some $\gamma \in \mathcal{O}_1$, where $\operatorname{Norm}(\gamma) = N\ell^a$ for some $a \in \mathbb{Z}_{\geq 0}$, and $\delta \in \mathbb{Z}j + \mathbb{Z}ij$, where $i, j \in B_{p,\infty}$ are the elements such that $i^2 = -1$ and $j^2 = -p$. The core of the KLPT algorithm is then to lift δ to a pair $(\lambda, \delta') \in \mathbb{Z} \times \mathcal{O}_1$ such that $\delta' = \lambda\delta \bmod N$ and $\operatorname{Norm}(\delta') = \ell^b$ for some $b \in \mathbb{Z}_{\geq 0}$. Since I is generated by N and $\gamma\delta'$, the ideal $I\overline{\gamma\delta'}/N$ is equivalent to I and of norm ℓ^{a+b} as desired. Note that the KLPT algorithm is only heuristic, but that a variant of it was proven rigorously assuming only GRH by Wesolowski [Wes22b].

Let E_1, E_2 be supersingular elliptic curves with endomorphism rings $\mathcal{O}_1$ and $\mathcal{O}_2$, respectively. Then the set of isogenies from E_1 to E_2 is a left $\mathcal{O}_1$-module and a right $\mathcal{O}_2$-module. In particular, they form a $\mathbb{Z}$-lattice of rank 4 [Voi21, Lem. 42.1.11]. By the following lemma, the $\mathbb{Z}$-lattice is isomorphic to a connecting left ideal I as an $\mathcal{O}_1$-module.

Lemma 5.2.2. *[Voi21, 42.2.8] Let $\operatorname{Hom}(E_2, E_1)$ denote the set of isogenies from E_2 to E_1 and let $\mathcal{O}_1$ and $\mathcal{O}_2$ denote the endomorphism rings of E_1 and E_2, respectively. Let I be a connecting ideal of $\mathcal{O}_1$ and $\mathcal{O}_2$ and let $\phi_I : E_2 \to E_1$ denote the corresponding isogeny. Then the map $\phi_I^* : \operatorname{Hom}(E_1, E_2) \to I$, $\psi \mapsto \psi \circ \phi_I$ is an isomorphism of left $\mathcal{O}_1$-modules.*

Since the KLPT algorithm computes a connecting ideal between two maximal orders, Lemma 5.2.2 implies that one can compute a $\mathbb{Z}$-basis of $\operatorname{Hom}(E_1, E_2)$. However, the degree of these isogenies might not be smooth and it is not obvious that one can evaluate them efficiently. In Algorithm 5.1, we will show that one can evaluate these isogenies on points efficiently using the KLPT algorithm.

5.2.2 LLL lattice reduction

Next, we recall some basic facts about lattice reduction, which aims to transform an arbitrary input basis into a basis of "higher quality". In the following, we are interested in bases that are close to orthogonal.

Let $B := (b_1, \ldots, b_n)$ be the basis of a lattice L, let π_i denote the projection onto $\operatorname{span}(b_1, \ldots, b_{i-1})$ for $i = \{1, \ldots, n\}$ and let $B^* := (b_1^*, \ldots, b_n^*)$ be the *Gram–Schmidt orthogonalisation* of B, where $b_i^* = \pi_i(b_i)$. Intuitively speaking, a good basis is one in which the sequence of Gram–Schmidt norms $\|b_1^*\|, \|b_2^*\|, \ldots, \|b_n^*\|$ does not decay too fast.

The Lenstra–Lenstra–Lovász (LLL) reduction calculates a short and nearly orthogonal lattice basis for any lattice in polynomial time [LLL82]. We recall a more precise statement in the following proposition using the Gram–Schmidt coefficients $\mu_{i,j} := \frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}$.

Proposition 5.2.3. *The LLL lattice reduction with factors (η, δ), where $\delta \in (0.25, 1)$ and $\eta \in [0.5, \sqrt{\delta}]$, provides in polynomial time a basis $B = (b_1, \ldots, b_n)$ that is size-reduced with $\mu_{i,j} < \eta$ for all $j < i$ and has Gram–Schmidt orthogonalisation satisfying the so-called Lovász condition $\delta \|b_i^*\|^2 \leq \|\mu_{i+1,i} b_i + b_{i+1}^*\|^2$.*

The default parameters for LLL reduction in MAGMA, which we use for the experiments later in this chapter, are $\delta = 0.75$ and $\eta = 0.501$. Since LLL-reduced bases are in some sense close to orthogonal, we can expect short vectors in the lattice to have rather small coefficients with respect to the basis. This is captured by the following lemma which is a consequence of [LLL82, Eq. (1.8)] and Cramer's rule.

Lemma 5.2.4. *Let L be a full rank lattice with LLL-reduced basis $b_1, \ldots, b_n$ with factors (η, δ) and let $v := \sum_{i=1}^{n} \gamma_i b_i \in L$. Then*

$$|\gamma_i| \leq \left(\frac{4}{(4\delta - 1)} \right)^{n(n-1)/4} \frac{|v|}{|b_i|}.$$

Proof. By [LLL82, Eq. (1.8)], an LLL-reduced basis $b_1, \ldots, b_n$ satisfies

$$\prod_{i=1}^{n} |b_i| \leq \left(\frac{4}{(4\delta - 1)} \right)^{n(n-1)/4} \det(L).$$

Using Cramer's rule, we therefore get

$$|\gamma_i| = \frac{\det(b_1, \ldots, b_{i-1}, v, b_{i+1}, \ldots, b_n)}{\det(L)} \leq \frac{|b_1| \cdots |b_{i-1}| \cdot |v| \cdot |b_{i+1}| \cdots |b_n|}{\det(L)} \cdot \frac{|b_i|}{|b_i|}$$
$$\leq \left(\frac{4}{(4\delta - 1)}\right)^{n(n-1)/4} \cdot \frac{|v| \cdot \det(L)}{|b_i| \cdot \det(L)} = \left(\frac{4}{(4\delta - 1)}\right)^{n(n-1)/4} \cdot \frac{|v|}{|b_i|}. \qquad \square$$

5.2.3 The reduction by GPST

In [GPST16, §4], Galbraith, Petit, Shani and Ti give an efficient reduction of computing the secret isogeny of an SIDH instance to the problem of computing the endomorphism rings of both the isogeny's domain and the codomain. We summarise their results and we recall why the algorithm does not work as such outside of the SIDH setting.

Let $\varphi : E_1 \to E_2$ be a ℓ^n-degree isogeny one wishes to recover given the two endomorphism rings $\mathcal{O}_1 = \mathrm{End}(E_1)$ and $\mathcal{O}_2 = \mathrm{End}(E_2)$. Since E_1 and E_2 are supersingular curves, their endomorphism rings are maximal orders in the quaternion algebra $B_{p,\infty}$. By Lemma 5.2.1, one can recover an ideal connecting $\mathcal{O}_1$ and $\mathcal{O}_2$ efficiently. Such an ideal corresponds to one of infinitely many isogenies between E_1 and E_2. This isogeny is in general not of degree ℓ^n and, in particular, it is not the same as φ. Yet, to attack SIDH, the isogeny needs to be of the correct degree and must have the correct action on the torsion points.

The secret isogenies in SIDH are of degree approximately $\sqrt{p}$. However, a pair of random supersingular elliptic curves over $\mathbb{F}_{p^2}$ is unlikely to be connected by an isogeny of degree significantly smaller than $\sqrt{p}$. In [GPST16], the authors leverage this observation to recover the target isogeny given the endomorphism rings of E_1 and E_2 as follows.

Given a connecting ideal I for the endomorphism rings, the authors compute a Minkowski reduced basis which is used to recover an element $\alpha \in I$ of minimal norm.

By [KLPT14, Lem. 5], the ideal $I' := I\overline{\alpha}/\mathrm{Norm}(I)$ is another ideal connecting $\mathcal{O}_1$ and $\mathcal{O}_2$ of (minimal) norm $\mathrm{Norm}(\alpha)$. Then, the isogeny $E_1 \to E_2$ of degree $\mathrm{Norm}(\alpha)$ corresponding to this ideal can be computed using Vélu's formulae. If the shortest isogeny between E_1 and E_2 is indeed of degree ℓ^n, this algorithm allows to recover such an isogeny of correct degree from the endomorphisms. The experimental results in [GPST16] suggest that, by trying relatively few small elements α in the previous algorithm, one recovers an isogeny that can be used to attack SIDH with overwhelming probability.

Clearly, the approach outlined above relies crucially on the fact that the degree of the sought isogeny is among the smallest possible degrees of isogenies connecting E_1

and E_2. In schemes that do not use secret isogenies of relatively small degree (e.g., B-SIDH [Cos20] or SÉTA [DDF+21]), the approach is infeasible.

5.3 Reducing isogeny finding to endomorphism ring computation

In this section, we describe an algorithm to evaluate non-smooth degree isogenies; and an algorithm to compute a secret isogeny $\phi : E_1 \to E_2$ of degree N_1 between supersingular elliptic curves, provided that certain N_2-torsion images and the endomorphism rings of E_1 and E_2 are known.

5.3.1 Evaluating non-smooth degree isogenies

In this subsection, we provide an algorithm solving the following task.

Task 5.3.1. *Let E_1 and E_2 be two curves with given endomorphism rings $\mathcal{O}_1$ and $\mathcal{O}_2$ respectively. Let I be an $\mathcal{O}_1$-left and $\mathcal{O}_2$-right ideal and let $P \in E_1$. Evaluate $\phi_I(P)$, where ϕ_I is the isogeny corresponding to the ideal I.*

Remark 5.3.2. The isogeny ϕ_I corresponding to the left ideal I is only unique up to post-composition with isomorphisms. Here, E_2 is a prescribed curve so the only potential issues arise from automorphisms of E_2. The number of automorphisms of E_2 can be bounded by a constant (in most cases it is actually 2), so there is some slight ambiguity in the end result of Task 5.3.1 which will eventually result in a constant overhead every time this subroutine is called.

To solve this task, we extend an algorithm due to Petit and Lauter [PL17, Alg. 3] which evaluates endomorphisms. Note that a solution to Task 5.3.1 evaluates isogenies of not necessarily smooth degree between curves with known endomorphism rings.

Petit–Lauter Algorithm [PL17, Alg. 3]

Let $(E_1, \mathcal{O}_1)$ denote a supersingular curve and its endomorphism ring, and let $w \in \mathcal{O}_1$. In order to evaluate the endomorphism $\phi_{w\mathcal{O}_1}$ on a point $P \in E_1$, the algorithm by Petit and Lauter uses a curve $(E_0, \mathcal{O}_0)$ whose endomorphisms can be efficiently evaluated, e.g. the curve with j-invariant 1728 (see Example 2.2.25). The algorithm proceeds as follows.

Let $\{w_1, w_2, w_3, w_4\}$ be a basis of $\mathcal{O}_0$ and let $\{\phi_1, \phi_2, \phi_3, \phi_4\}$ be the corresponding basis of $\mathrm{End}(E_0)$. The core idea of the algorithm is to use the KLPT algorithm to compute a powersmooth isogeny $\varphi : E_1 \to E_0$ of degree N.

Then, we have $N\mathcal{O}_1 \subset \mathcal{O}_0$ and thus $Nw \in \mathcal{O}_0$. For $w = (a_1w_1+a_2w_2+a_3w_3+a_4w_4)/N$, this implies

$$\phi_{w\mathcal{O}_1} = \varphi^{-1} \circ \frac{a_1\phi_1 + a_2\phi_2 + a_3\phi_3 + a_4\phi_4}{N} \circ \varphi,$$

where $\varphi^{-1} := \frac{1}{\deg\varphi}\widehat{\varphi}$. Since all the isogenies on the right-hand side can be evaluated efficiently, this allows to evaluate $\phi_{w\mathcal{O}_1}$.

Solving Task 5.3.1:

Let $(E_1, \mathcal{O}_1)$ and $(E_2, \mathcal{O}_2)$ be supersingular elliptic curves with their endomorphism rings, let I be an $\mathcal{O}_1$-left and $\mathcal{O}_2$-right ideal of non-smooth norm and let $P \in E_1$. We would like to evaluate the isogeny ϕ_I corresponding to the ideal I at the point P.

Using the KLPT algorithm, we compute an $\mathcal{O}_1$-right and $\mathcal{O}_2$-left ideal J whose smooth norm is coprime to that of I. Then, the ideal IJ represents an endomorphism $w \in \mathcal{O}_1$ of E_1. The element $w \in \mathcal{O}_1$ can be recovered by computing the shortest vector in IJ. We obtain $IJ = w\mathcal{O}_1$ for some $w \in \mathcal{O}_1$. Using [PL17, Alg. 3], we evaluate $Q := \phi_{w\mathcal{O}_1}(P)$, and compute $\phi_I(P) := \phi_J^{-1}(Q)$. We summarise the steps in Algorithm 5.1.

Algorithm 5.1: Evaluating non-smooth degree isogenies

Input: Elliptic curves E_1, E_2 with endomorphism rings $\mathcal{O}_1, \mathcal{O}_2$ and an $\mathcal{O}_1$-left and $\mathcal{O}_2$-right ideal I together with a point $P \in E_1$, an elliptic curve E_0 such that its endomorphism ring $\mathcal{O}_0$ with basis $\{w_1, w_2, w_3, w_4\}$ corresponds to endomorphisms $\phi_1, \phi_2, \phi_3, \phi_4$ that can be evaluated efficiently.

Output: $\phi_I(P)$.

1 Compute an $\mathcal{O}_1$-right and $\mathcal{O}_2$-left ideal J whose smooth norm is coprime to that of I using Wesolowski's algorithm [Wes22b] (or KLPT).

2 Compute an $\mathcal{O}_1$-left and $\mathcal{O}_0$-right ideal K of powersmooth norm N using Wesolowski's algorithm (or KLPT).

3 Set $IJ = w\mathcal{O}_1$ for some $w \in \mathcal{O}_1$ and find integers a_1, a_2, a_3 and a_4 such that $Nw = a_1w_1 + a_2w_2 + a_3w_3 + a_4w_4$.

4 Evaluate $Q := \phi_{IJ}(P) = \frac{\phi_K^{-1} \circ (a_1\phi_1+a_2\phi_2+a_3\phi_3+a_4\phi_4) \circ \phi_K(P)}{N}$ using [PL17, Alg. 3].

5 **return** $\phi_I(P) := \phi_J^{-1}(Q)$.

Lemma 5.3.3. *Assuming GRH, Algorithm 5.1 runs in polynomial time.*

Proof. The endomorphism rings of the curves E_0, E_1 and E_2 are known. For this case, Wesolowski gave a polynomial-time algorithm to compute a connecting smooth ideal in

polynomial time assuming only GRH [Wes22b]. Previously, a similar (faster) polynomial-time algorithm, KLPT [KLPT14], was already known for this task, but it relies on heuristics. Thus, Steps 1 and 2 run in polynomial time.

The ideal I ($\mathcal{O}_1$-left and $\mathcal{O}_2$-right) and J ($\mathcal{O}_1$-right and $\mathcal{O}_2$-left) have coprime norms, hence the two-sided $\mathcal{O}_1$ ideal IJ corresponds to a non trivial endomorphism $w \in \mathcal{O}_1$ of E_1 that can be recovered by computing a Minkowski reduced basis of IJ. For lattices up to dimension 4, a Minkowski reduced basis can be computed in polynomial time [NS04]. The integers a_1, a_2, a_3 and a_4 are obtained by rewriting the quaternion Nw as an element of $\mathcal{O}_0$. Therefore, Step 3 runs in polynomial time. By hypothesis, the isogenies ϕ_1, ϕ_2, ϕ_3 and ϕ_4 can be evaluated efficiently. The ideals K and J have smooth norm, hence the isogenies ϕ_K, ϕ_K^{-1} and ϕ_J^{-1} have smooth degree and can also be evaluated efficiently for example using Vélu's formulae. It follows that Step 4 and Step 5 run in polynomial time as well. □

5.3.2 Computing isogenies using torsion information

As recalled in Section 5.2.3, [GPST16] gives a strategy to compute an isogeny ϕ between two curves E_1 and E_2 with known endomorphism rings $\mathcal{O}_1$ and $\mathcal{O}_2$, if its degree is minimal (i.e., ϕ is an isogeny of minimal degree connecting E_1 and E_2). The algorithm in [GPST16] applies to the SIDH setting where the degree of the secret isogeny is minimal with non-negligible probability (or otherwise at least of particularly small degree). Meanwhile, the torsion point information available in SIDH-like schemes is not used at all.

In this section, we will generalise this algorithm. We will show how the torsion point information in SIDH-like schemes can be exploited together with the knowledge of endomorphism rings to compute a secret isogeny of arbitrary (fixed) degree, if it exists.

The strategy is as follows. Let $\phi : E_1 \to E_2$ be a secret isogeny, let P, Q be a basis of $E_1[N_2]$ and let $\phi(P)$, $\phi(Q)$ be the torsion information provided in SIDH-like schemes. Let $I(\mathcal{O}_1, \mathcal{O}_2)$ be a connecting ideal between the maximal orders $\mathcal{O}_1$ and $\mathcal{O}_2$. Instead of solving for a minimal norm element of the ideal $I(\mathcal{O}_1, \mathcal{O}_2)$ as in [GPST16], we compute an LLL-reduced basis $\{\psi_1, \psi_2, \psi_3, \psi_4\}$ of I.

Using Algorithm 5.1, the isogenies ψ_i, $i = 1, \ldots, 4$, can be evaluated at the points P and Q. Next, we want to write ϕ in terms of our LLL-reduced basis, i.e. we want to find $(x_1, \ldots, x_4) \in \mathbb{Z}^4$ such that

$$\phi = x_1\psi_1 + x_2\psi_2 + x_3\psi_3 + x_4\psi_4. \tag{5.1}$$

Clearly, recovering x_i allows us to compute the secret isogeny ϕ. Note that Eq. (5.1) implies in particular

$$\sum_{i=1}^{4} x_i\psi_i(P) = \phi(P) \qquad \text{and} \qquad \sum_{i=1}^{4} x_i\psi_i(Q) = \phi(Q). \tag{5.2}$$

To compute x_1, x_2, x_3 and x_4, we first prove that a solution to Eq. (5.2) is unique modulo N_2. Then, we use simple linear algebra methods to recover it. Finally, we will show that knowing the x_i modulo N_2 is enough to recover them exactly (as integers).

Lemma 5.3.4. *Let E_1, E_2 be supersingular elliptic curves over $\mathbb{F}_{p^2}$ and let P, Q be a basis of $E_1[N_2]$. Let $\psi_1, \psi_2, \psi_3, \psi_4$ be a $\mathbb{Z}$-basis of $\mathrm{Hom}(E_1, E_2)$. The system of linear equations modulo N_2 corresponding to*

$$\sum_{i=1}^{4} x_i\psi_i(P) = \phi(P) \quad \textit{and} \quad \sum_{i=1}^{4} x_i\psi_i(Q) = \phi(Q)$$

has a unique solution $(x_1, x_2, x_3, x_4) \in (\mathbb{Z}/N_2\mathbb{Z})^4$.

Proof. Let P', Q' be a basis of $E_2[N_2]$. Every isogeny ϕ in $\mathrm{Hom}(E_1, E_2)$ can be identified with a matrix $\begin{pmatrix} a & b \\ c & d \end{pmatrix} \in M_2(\mathbb{Z}/N_2\mathbb{Z})$ by writing its images on $E_1[N_2]$ as

$$\phi(P) = [a]P' + [c]Q', \quad \phi(Q) = [b]P' + [d]Q'.$$

Let $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ be a matrix in $M_2(\mathbb{Z}/N_2\mathbb{Z})$. First, we prove that for any matrix A, there exists an isogeny $\phi \in \mathrm{Hom}(E_1, E_2)$ such that representation of ϕ is A.

Let $\psi : E_1 \to E_2$ be an isogeny such that the degree of ψ is coprime to N_2. Note that such an isogeny exists as the ℓ-isogeny graph is connected for any prime ℓ. Let M be the matrix corresponding to ψ. Since the degree of ψ is coprime to N_2, it corresponds to an invertible matrix in $M_2(\mathbb{Z}/N_2\mathbb{Z})$.

It is known (see [Voi21, Thm. 42.1.9]) that $\mathrm{End}(E_1)/N_2\,\mathrm{End}(E_1)$ is isomorphic to $M_2(\mathbb{Z}/N_2\mathbb{Z})$ (the injection is clear, surjectivity is the key result). Note that the isomorphism depends on a choice of basis of $E_1[N_2]$. Consider the isomorphism corresponding to the basis P, Q. Then, there exists an endomorphism $\theta \in \mathrm{End}(E_1)$ whose matrix representation is AM^{-1}. This implies that the matrix representation of $\phi = \theta \circ \psi$ is $AM^{-1}M = A$, i.e. there exists an isogeny from E_0 to E_1 that is represented by the matrix A.

Clearly, $\sum_{i=1}^{4} x_i\psi_i$ and $\sum_{i=1}^{4} y_i\psi_i$ are represented by the same matrix if for $i = 1, \ldots, 4$ we have $x_i \equiv y_i \pmod{N_2}$. Thus, there are at most $N_2^4 = |(\mathbb{Z}/N_2\mathbb{Z})^4|$ different matrices that one can obtain.

Now, the lemma follows by a simple counting argument. Since every matrix in $M_2(\mathbb{Z}/N_2\mathbb{Z})$ is represented for an isogeny, every matrix must uniquely correspond to a sum of the form $\sum_{i=1}^{4} x_i\psi_i$ modulo N_2. Consequently, if a matrix has two different representations of the form $\sum_{i=1}^{4} x_i\psi_i$, then they are the same modulo N_2. □

Remark 5.3.5. The main result of the proof is that $\mathrm{Hom}(E_1, E_2)$ modulo N_2 is isomorphic to $M_2(\mathbb{Z}/N_2\mathbb{Z})$ as a $\mathbb{Z}/N_2\mathbb{Z}$-module [Tat66]. Informally, the key idea is that $\mathrm{Hom}(E_1, E_2)$ is a left ideal in $\mathrm{End}(E_1)$, hence it will be a left ideal in $M_2(\mathbb{Z}/N_2\mathbb{Z})$. Since isogenies between E_1 and E_2 of degree coprime to N_2 exist, this left ideal will contain invertible matrices, hence it must be the entire matrix ring.

Next, we provide details on how to recover x_1, x_2, x_3, x_4. Let $\{\psi_1, \psi_2, \psi_3, \psi_3\}$ be our LLL-reduced basis of $\mathrm{Hom}(E_1, E_2)$. Given $\psi_i(P)$, $\psi_i(Q)$ for $i = 1, 2, 3, 4$ and $\phi(P)$, $\phi(Q)$, we would like to compute $(x_1, \cdots, x_4) \in (\mathbb{Z}/N_2\mathbb{Z})^4$ such that

$$\sum_{i=1}^{4}[x_i]\psi_i(P) = \phi(P) \qquad \text{and} \qquad \sum_{i=1}^{4}[x_i]\psi_i(Q) = \phi(Q).$$

Note that N_2 is a smooth integer and that $\phi(P)$ and $\phi(Q)$ form a basis of $E_2[N_2]$ as $\deg(\phi)$ and N_2 are coprime. For $i = 1, 2, 3, 4$, we can compute the integers $a_i, b_i, c_i, d_i \in \mathbb{Z}/N_2\mathbb{Z}$ such that $\psi_i(P) = [a_i]\phi(P) + [b_i]\phi(Q)$ and $\psi_i(Q) = [c_i]\phi(P) + [d_i]\phi(Q)$ by using the Weil pairing and solving discrete logarithms in a group of smooth order. Now, the integers $(x_1, \cdots, x_4) \in (\mathbb{Z}/N_2\mathbb{Z})^4$ satisfy

$$\phi(P) = \left[\sum_{i=1}^{4} x_i a_i\right]\phi(P) + \left[\sum_{i=1}^{4} x_i b_i\right]\phi(Q)$$

and

$$\phi(Q) = \left[\sum_{i=1}^{4} x_i c_i\right]\phi(P) + \left[\sum_{i=1}^{4} x_i d_i\right]\phi(Q).$$

We obtain

$$\begin{pmatrix}1 & 0 & 0 & 1\end{pmatrix} = \begin{pmatrix}x_1 & x_2 & x_3 & x_4\end{pmatrix} \cdot \begin{pmatrix}a_1 & b_1 & c_1 & d_1\\ a_2 & b_2 & c_2 & d_2\\ a_3 & b_3 & c_3 & d_3\\ a_4 & b_4 & c_4 & d_4\end{pmatrix}.$$

By Lemma 5.3.4, there exists a unique solution $\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \end{pmatrix}$ to the previous equation. Hence the matrix

$$M := \begin{pmatrix} a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ a_3 & b_3 & c_3 & d_3 \\ a_4 & b_4 & c_4 & d_4 \end{pmatrix}$$

is invertible and the solution is given by $\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 1 \end{pmatrix} \cdot M^{-1}$. The latter operation corresponds to adding the first and the fourth row of M^{-1}. We summarise this process in Algorithm 5.2.

Algorithm 5.2: Computing a solution to the linear system

Input: $\psi_i(P)$ and $\psi_i(Q)$ for $i = 1, \ldots, 4$, where ψ_i are a $\mathbb{Z}$-basis of $\mathrm{Hom}(E_1, E_2)$; $\phi(P)$ and $\phi(Q)$ of smooth order N_2.

Output: x_1, x_2, x_3, x_4 such that $\sum_{i=1}^{4}[x_i]\psi_i(P) = \phi(P)$, and $\sum_{i=1}^{4}[x_i]\psi_i(Q) = \phi(Q)$.

1 **for** $i = 1, \cdots, 4$ **do**

2 Compute $a_i, b_i, c_i, d_i \in \mathbb{Z}/N_2\mathbb{Z}$ such that $\psi_i(P) = [a_i]\phi(P) + [b_i]\phi(Q)$ and $\psi_i(Q) = [c_i]\phi(P) + [d_i]\phi(Q)$.

3 Set M to be the 4×4 matrix whose rows are $\begin{pmatrix} a_i & b_i & c_i & d_i \end{pmatrix}$ for $i = 1, 2, 3, 4$.

4 Compute the inverse matrix M^{-1} of M.

5 Set $\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \end{pmatrix}$ to be the sum of the first and the fourth rows of M^{-1}.

6 **return** x_1, x_2, x_3, x_4 *such that* $|x_i| < N_2/2$.

Lemma 5.3.6. *Algorithm 5.2 is correct and runs in polynomial time provided that N_2 is smooth.*

Proof. Follows from the previous discussion. □

The following Lemma 5.3.7 gives a condition under which the solution computed in Algorithm 5.2 yields a solution to Eq. (5.1).

Lemma 5.3.7. *Let $d := \min\{\deg(\varphi) \mid \varphi : E_1 \to E_2$ is isogeny$\}$ and $\frac{N_1}{N_2} < \frac{d}{16}$. Given the solution $x_1, \ldots, x_4$ to the system of linear equations modulo N_2 returned by Algorithm 5.2, where $\sum_{i=1}^{4}[x_i]\psi_i(P) = \phi(P)$, $\sum_{i=1}^{4}[x_i]\psi_i(Q) = \phi(Q)$, we have $\phi = \sum_{i=1}^{4}[x_i]\psi_i$ in $\mathrm{Hom}(E_1, E_2)$.*

Proof. By Lemma 5.2.4, with $\delta = 0.75$ and $n = 4$, we have that $\phi = \sum_{i=1}^{4}[\gamma_i]\psi_i$ where $|\gamma_i| \leq \frac{8\deg(\phi)}{\deg(\psi_i)} \leq \frac{8N_1}{d}$. It follows that $|\gamma_i| \leq \frac{8N_1}{d} < \frac{N_2}{2}$ since $\frac{N_1}{N_2} < \frac{d}{16}$ by hypothesis.

The solution $(x_1,\ x_2,\ x_3,\ x_4)$ returned by Algorithm 5.2 satisfies $|x_i| < \frac{N_2}{2}$ for $i = 1, 2, 3, 4$. Moreover, by Lemma 5.3.4, this solution is unique modulo N_2. Thus, $\phi = \sum_{i=1}^{4}[x_i]\psi_i$ in $\mathrm{Hom}(E_1, E_2)$. □

The entire process of computing isogenies of a specific but arbitrary degree between two supersingular curves with known endomorphism ring is summarised in Algorithm 5.3.

Algorithm 5.3: Computing isogeny with torsion point information

Input: Supersingular elliptic curves E_1, E_2 with known endomorphism rings $\mathcal{O}_1, \mathcal{O}_2$ which are connected by an isogeny ϕ of degree N_1 and $\phi(P), \phi(Q)$, where P, Q are a basis of $E_1[N_2]$, such that $\frac{N_1}{N_2} < \frac{d}{16}$.

Output: ϕ.

1 Compute a basis of an $\mathcal{O}_1$-left and $\mathcal{O}_2$-right ideal I.
2 Compute an LLL-reduced basis $\psi_1, \psi_2, \psi_3, \psi_4$ of I.
3 Compute $\psi_i(P), \psi_i(Q)$ using Algorithm 5.1.
4 Use Algorithm 5.2 to solve for $|x_i| < N_2/2$ such that $\sum_{i=1}^{4}[x_i]\psi_i(P) = \phi(P),\ \sum_{i=1}^{4}[x_i]\psi_i(Q) = \phi(Q)$.
5 Compute isogeny from the relation $\phi = \sum_{i=1}^{4}[x_i]\psi_i$.
6 **return** ϕ.

Finally, we prove that Algorithm 5.3 succeeds in polynomial time.

Theorem 5.3.8. *Let* $d := \min\{\deg(\phi) \mid \phi : E_1 \to E_2 \text{ is isogeny}\}$. *Assuming GRH, Algorithm 5.3 solves Task 5.1.1 in polynomial time, whenever* $\frac{N_1}{N_2} < \frac{d}{16}$.

Proof. Correctness of the algorithm follows from Lemma 5.3.7 and the preceding discussion. We are left to show the polynomial running time. Step 1 can be computed using an efficient algorithm due to Kirschmer and Voight [KV10] (or using the KLPT algorithm [KLPT14], but we do not need the connecting ideal to have a smooth norm). Step 2 is the LLL lattice reduction algorithm for a four-dimensional lattice which also runs in polynomial time. Step 3 and Step 4 run in polynomial time by Lemma 5.3.3 and Lemma 5.3.6, respectively. □

Remark 5.3.9. The condition $\frac{N_1}{N_2} < \frac{d}{16}$ could be weakened to $\frac{N_1}{N_2} \leq \frac{d}{16}$ in which case we get that $|x_i| \leq N_2/2$. However, when N_2 is even and x_i is congruent to $N_2/2$, then the lift to the above range is not unique (as $-N_2/2$ and $N_2/2$ represent the same residue class). This is not an issue for Algorithm 5.3 as one will have multiple candidates for ψ (16 of them in the worst case) that can be tested. By looking at the degrees, the correct one can be chosen efficiently. More generally, one could relax the statement of Theorem 5.3.8 further by allowing non-unique lifts and adding an additional step to check for the correctness of solutions at the end of Algorithm 5.3.

Remark 5.3.10. As was shown in Lemma 5.3.7, Algorithm 5.3 requires an amount of torsion point information that depends on the degree d of the shortest isogeny between the supersingular elliptic curves E_1 and E_2.

Many applications of cryptographic interest use balanced parameters where $N_1 \approx N_2$. Taking $\frac{N_1}{N_2} \approx 1$, the procedure above works whenever the two curves are not connected by an isogeny of degree smaller than 16. This can be checked easily with an exhaustive search.

Remark 5.3.11. Algorithm 5.3 does not use the fact that N_1 is smooth. If one wants to retrieve the secret isogeny as a rational map (as a composition of small degree maps), then clearly the smoothness of N_1 is still required. However, if one only wants to evaluate the secret isogeny at any point coprime to its degree (e.g. as in pSIDH [Ler22]), then this can be accomplished by Algorithm 5.3 even if N_1 is not smooth.

5.3.3 Computational example

We illustrate our reduction with an example.

Consider the prime $p = 83701957499$, where we have $p + 1 = 2^2 \cdot 3^{14} \cdot 5^4 \cdot 7$. Let $B_{p,\infty}$ be the quaternion algebra ramified at p and ∞ and generated over the rationals by i, j, k, where $i^2 = -1$, $j^2 = -p$, and $k = ij$. Fix the finite field $\mathbb{F}_{p^2}$ where $\alpha^2 = -1$ generates $\mathbb{F}_{p^2}$ over $\mathbb{F}_p$. Take the elliptic curve given by $E_0 : y^2 = x^3 + x$ which has j-invariant 1728 and endomorphism ring generated by

$$1, i, \frac{1+ij}{2}, \frac{i+j}{2}$$

as was described in Example 2.2.25. Let the secret isogeny be a 3^{14}-isogeny $\theta : E_0 \to E$ and let the torsion point images of $E_0[5^4]$ under θ be known, i.e. $\theta_{|E_0[5^4]}$ is known. For the purpose of illustrating the reduction, we use the *secret* θ to recover the endomorphism ring of E which is generated by

$$\frac{5159993 + j + 10319986i + 11800766447346k}{9565938}, \frac{2j + 6291065i + 7411685041437k}{9565938}, \frac{3i + 196249k}{2}, 1594323k.$$

However, note that this computation involving the secret only computes the endomorphism ring which we assume to be already known for our reduction.

Now, using the knowledge of both endomorphism rings, our reduction proceeds as follows. First, we compute a connecting ideal between the two endomorphism rings and also compute a reduced basis of the ideal

$$\frac{227049 + j + 154612i}{2}, \frac{154612 - 227049i + k}{2}, \frac{121127 - 9j + 4995744i + 14k}{2}, \frac{4995744 - 14j - 121127i - 9k}{2}.$$

Interpreting these quaternions as endomorphisms, we can map generators of $E_0[5^4]$ through them. We fix the following generators of $E_0[5^4]$

$$\begin{aligned} P_5 &= (75854242840\alpha + 62002351922, 51107649030\alpha + 19190692821), \\ Q_5 &= (17857458337\alpha + 504604508, 77775481527\alpha + 25718537048). \end{aligned}$$

In particular, by naming the endomorphisms corresponding to the reduced basis elements $\psi'_1, \psi'_2, \psi'_3, \psi'_4$, respectively, we have that

$$\begin{aligned} \psi'_1(P_5) &= (9049577476\alpha + 26838535531, 9532248787\alpha + 18861270144)\,, \\ \psi'_1(Q_5) &= (14085392798\alpha + 75272963133, 35152660085\alpha + 3705843319)\,, \\ \psi'_2(P_5) &= (54148936824\alpha + 29574813, 27904476482\alpha + 79581351851)\,, \\ \psi'_2(Q_5) &= (6218706354\alpha + 14437916419, 19897519544\alpha + 26853032937)\,, \\ \psi'_3(P_5) &= (27253519435\alpha + 63921648196, 55371710596\alpha + 3587102479)\,, \\ \psi'_3(Q_5) &= (6221393886\alpha + 23453138168, 81414672111\alpha + 63571818133)\,, \\ \psi'_4(P_5) &= (20904892135\alpha + 45099774747, 32347928248\alpha + 14718113311)\,, \\ \psi'_4(Q_5) &= (16837240041\alpha + 11444980635, 5815630261\alpha + 82050564219)\,. \end{aligned}$$

Furthermore, we know the images of P_5 and Q_5 through the secret isogeny θ. Note that these ψ'_i are not the same as the ψ_i defined in the previous section as they are endomorphisms of E_0. However, they are just the original ψ_i composed with the isogeny between E and E_0 coming from KLPT. We will denote the actual isogenies corresponding to them by ψ_i. They can be evaluated at P_5 and Q_5 by applying the connecting isogeny to them and multiplying it with the inverse of its degree modulo 5^4. These are points in E, and in particular, they are in the subgroup $E[5^4]$. This allows us to express them in terms of $\theta(P_5)$ and $\theta(Q_5)$ which we were given.

This results in the following 4×4 matrix

$$\begin{pmatrix} 222 & 128 & 484 & 474 \\ 311 & 363 & 337 & 12 \\ 184 & 477 & 307 & 574 \\ 344 & 566 & 191 & 132 \end{pmatrix}$$

whose first row represents the four coefficients that express $\psi_1(P_5)$ as a linear combination of $\theta(P_5)$ and $\theta(Q_5)$, and $\psi_1(Q_5)$ as a linear combination of $\theta(P_5)$ and $\theta(Q_5)$. For example,

$$\psi_2(Q_5) = [337]\theta(P_5) + [12]\theta(Q_5).$$

Inverting this matrix and summing the first and fourth rows allow us to recover the coefficients x_i's providing the expression of the secret isogeny as a linear combination of ψ_1, ψ_2, ψ_3 and ψ_4. The result of the computation is that

$$\theta = [14]\psi_1 + [9]\psi_2 + \psi_4.$$

One can verify that this is correct. Note that this verification can be done without computing the ψ_i but by computing that the degree of this linear combination is indeed 3^{14} (as the action on the 5^4-torsion is already correct).

Remark 5.3.12. The secret isogeny in this example is not the isogeny between E_0 and E of smallest degree, hence the algorithm from [GPST16] would not have been sufficient to find θ.

5.4 Reduction in the presence of countermeasures against SIDH attacks

After publication of the results of this chapter up to this point at PKC 2022 [FKMT22], several papers emerged that broke SIDH and SIDH-based schemes such as B-SIDH in classical polynomial time [CD22, MM22, Rob22a]. In this section, we will argue that even in the presence of countermeasures proposed against these recent attacks, our reduction still applies.

First, observe that our reduction works in a more general context. Namely, [CD22] and [MM22] require the cardinality of the subgroup generated by the known torsion point images to be at least as large as the degree of the secret isogeny, i.e. $N_1 \approx N_2$. Robert's attack only requires $N_1 < N_2^2$ [Rob22a]. For B-SIDH parameters where the secret isogeny is of degree p this matches roughly the amount of torsion point information required for our reduction to work, as we expect the shortest connecting isogeny to be of degree roughly $\sqrt{p}$. However, for the general SSI-T problem it is easy to construct parameter sets for which the requirements of our attacks are weaker than what is required by Robert's attack. For an example, consider the case where $N_1 \approx p^{3/4}$. This is a case not covered by [GPST16] and the polynomial attack by Robert requires that

$N_2 \approx p^{3/8}$ [Rob22a]. For $p \approx N_1 N_2$, our reduction still works whenever $N_2 \approx p^{1/4}$, which follows from Theorem 5.3.8 and the fact that the shortest isogeny between two supersingular elliptic curve is roughly of degree $\sqrt{p}$. Note that the shortest isogeny does not need to lie in one ℓ-isogeny graph for a fixed ℓ but rather lies in the union of all ℓ-isogeny graphs.

The second remark concerns countermeasures. Since the attacks were published, two countermeasures have been proposed. In [Mor22], Moriya proposes to mask the degree of the secret isogeny. This prevents all previous attacks. Yet, our reduction still works as we never use N_1 explicitly and our reduction only requires an upper bound on the degree. A sufficient upper bound ($\sqrt{p}$ in SIDH and p in B-SIDH) is always known to the attacker.

Another countermeasure proposed by Fouotsa [Fou22] suggests to only reveal a secret multiple of the torsion point images (coprime to the order of the torsion points) as this information is sufficient to compute pushforwards of the secret isogenies. To prevent attackers from computing the secret multiple using pairings, one has to use a prime p such that $p+1$ is a product of many distinct small primes. Assume we are not given exact torsion point images $\varphi(P_B)$ and $\varphi(Q_B)$, but their multiples $[\lambda]\varphi(P_B)$ and $[\lambda]\varphi(Q_B)$ instead, where λ is a secret integer. In this case, Task 5.1.1 becomes the following.

Task 5.4.1. *Let N_1, N_2 be coprime integers and let $\varphi : E_1 \to E_2$ be a secret isogeny of degree N_1 between two supersingular elliptic curves. Let P_B, Q_B be a basis of $E_1[N_2]$. Given $\mathrm{End}(E_1)$, $\mathrm{End}(E_2)$, $[\lambda]\varphi(P_B)$, and $[\lambda]\varphi(Q_B)$ for an unknown $\lambda \in \mathbb{Z}$ coprime to N_2, find an isogeny $\varphi' : E_1 \to E_2$ of degree N_1 such that $\varphi_{|E_1[N_2]} = \varphi'_{|E_1[N_2]}$.*

For the rest of this section, we will describe how our reduction can be extended to solve this task.

Using Algorithm 5.2 for this task mutatis mutandis, the resulting system of equations will have 5 variables and 4 equations instead. By Lemma 5.3.4, this equation has rank 4. Hence, there will be one degree of freedom and every solution lies on a line in $(\mathbb{Z}/N_2\mathbb{Z})^5$. Thus, there are too many solutions for an exhaustive search. However, for slightly weaker bounds, we can still make our reduction work.

Theorem 5.4.2. *Let $d := \min\{\deg(\phi) \mid \phi : E_1 \to E_2 \text{ is isogeny}\}$. We can solve Task 5.4.1 in heuristic polynomial time whenever $\frac{N_1}{d} < \frac{{N_2}^{3/4}}{8}$.*

Proof. As in the proof of Lemma 5.3.4, denote by $\{\psi_1, \psi_2, \psi_3, \psi_3\}$ an LLL reduced basis for $\mathrm{Hom}(E_1, E_2)$. Recall the inequality $\phi = \sum_{i=1}^{4} \gamma_i \psi_i$ where $|\gamma_i| \leq \frac{8\deg(\phi)}{\deg(\psi_i)} \leq \frac{8N_1}{d}$. For $\frac{N_1}{d} < \frac{{N_2}^{3/4}}{8}$ this implies $|\gamma_i| < {N_2}^{3/4}$. Thus, one is looking for a very special solution to

the above homogeneous linear system. More precisely, the first 4 variables have to be smaller than ${N_2}^{3/4}$ and the last variable is smaller than N_2 (i.e., no condition is imposed on the last variable). This is an SIS-like lattice problem where the lattice consists of the integer solutions to this system of equations. The SIS problem stands for short integer solution and is a standard problem in lattice-based cryptography, which was first introduced by Ajtai [Ajt96]. The only difference is that one is not looking for a short vector with respect to the Euclidean metric but for a short vector in a slightly different metric. The volume of the defined rectangle is N_2^4, i.e. the size of the lattice determinant. Heuristically, we expect there to be a unique lattice vector in this rectangle. This vector can be found in two different ways: either using a weighted inner product which puts large weights on the first 4 variables and very little weight on the last variable or by orthogonal projection to the vector $(0, 0, 0, 0, N_2)$ and using general lattice reduction in the projected lattice. Being left with a closest vector problem (CVP) in dimension 5, the problem can be solved easily [HPS11]. □

5.5 Relevance to isogeny-based cryptography

In this chapter, we showed how to compute an isogeny of a specific degree between two supersingular elliptic curves, given their endomorphism rings and the images of some torsion points under the isogeny. We use this section to briefly summarise the impact of our results on different isogeny-based constructions.

For a long time, the isogeny-based community considered the meet in the middle attack (MITM) as the best attack when addressing the security level λ of isogeny-based schemes. Meanwhile, the MITM attack requires exponential storage, hence it may be unrealistic in practice. Therefore, more recently the van Oorschot–Wiener (vOW) parallel collision finding algorithm [vW99] was considered for the isogeny computation problem [ACC+19, CLN+20]. The vOW collision search allows for a space-time trade-off in the generic MITM, leading to a higher time complexity when limited storage is used.

Estimating the security level of isogeny-based schemes using vOW instead suggests that one can reduce the size of parameters that were previously fixed due to considering the generic MITM attack with unrealistic memory requirements. For B-SIDH, the proposed prime $p \approx N_1 N_2$ is roughly $2^{2\lambda}$. Given the analysis of the vOW collision search attack in [LWS21] to recover isogenies of degree N_1 or N_2, one may be tempted to propose smaller B-SIDH primes in order to improve B-SIDH's efficiency. Similar proposals were made for SIDH [LWS21].

However, we must also consider the attack in which the endomorphism rings of supersingular elliptic curves are computed so that Algorithm 5.3 of this chapter can be used to compute the secret isogeny. The complexity of this approach depends primarily on the size of the finite field $\mathbb{F}_{p^2}$ over which the curves are defined. As stated in Section 2.4, the classical and quantum complexity for computing endomorphism rings of supersingular elliptic curves is $O(\log(p)^2 p^{1/2})$ and $O^*(p^{1/4})$, respectively. Consequently, the parameter p must also satisfy $2^{2\lambda} < p$. Thus, lowering p below $2^{2\lambda}$ would make B-SIDH vulnerable to attacks that compute endomorphism rings and use the results of this chapter, showing that the size of p in B-SIDH was tight.

In the meantime certain attacks [CD22, MM22, Rob22a] broke SIDH, B-SIDH and SÉTA using the provided torsion point information and the degree of the secret isogenies. Note that the direct key recovery attack uses the algorithm to evaluate non-smooth isogenies presented in Section 5.3.1 as a subroutine [Wes22c].

Countermeasures which mask the degree of the secret isogeny and/or mask the available torsion point information have been proposed to thwart these attacks [Mor22, Fou22]. Schemes deploying these countermeasures are less efficient, but they are still considered secure at the time of writing this thesis. We showed in this chapter that our reduction still holds in the presence of said countermeasures. The conclusion that the prime p cannot be lowered therefore still applies to the versions of B-SIDH deploying the suggested countermeasures, i.e. this chapter's results provide an upper bound on the security of schemes like B-SIDH.

Furthermore, there are different ways of interpreting the results of this chapter. For example, proposing schemes with longer isogeny walks than the ones in B-SIDH does not provide any additional security benefit, as the attacker could then just compute endomorphism rings and use the reduction of this chapter to compute the secret isogeny. Perhaps this is not entirely unexpected given that the walks in B-SIDH have lengths which are comparable to the diameter of the supersingular ℓ-isogeny graph for a fixed ℓ. Finally, our results also imply that when (masked) torsion point images are provided, the problem of finding one isogeny between two supersingular elliptic curves becomes equivalent to finding an isogeny of a specific degree between these two supersingular curves, for a wide range of parameters.

CHAPTER 6

SCALLOP: Scaling the CSI-Fish

For all practical purposes, this chapter is identical to the following paper which introduces a new isogeny-based group action and was published as

Luca De Feo, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Simon-Philipp Merz, Lorenz Panny, and Benjamin Wesolowski. SCALLOP: Scaling the CSI-FiSh. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 345–375. Springer, Heidelberg, May 2023.

6.1 Introduction

A major breakthrough for isogeny-based group actions was the invention of the CSIDH key exchange [CLM+18]. 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, instead of ordinary curves, and this makes the scheme a lot faster for various reasons. This made CSIDH the first *efficient* post-quantum cryptographic group 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 [Pei20, BS20, CSCJR22].

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 [Sto12]. 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 [DG19] 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 remained impractical for the fastest parameter set (2 minutes), even after further optimisations by Decru, Panny and Vercauteren [DPV19].

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 [BKV19] 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 [BKV19]. CSI-FiSh is very efficient and is a building block for many other schemes such as threshold signatures [DM20, CS20], ring signatures [BKP20, LD21]) and group signatures [BDK+22]. Furthermore, it is a basis for other primitives such as updatable encryption [LR22].

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. The algorithm uses an index-calculus approach to compute the class group structure and instantiating CSI-FiSh for larger security levels of CSIDH would require class group computations that are currently firmly out of reach. Yet, especially in light of recent debates 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 cryptographic isogeny group actions that have better control on the class group. Thus, it is 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 [DDF+21]. 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 [CK19, DD22] and [ACL+22, Thm. 11.4]).

In this chapter, we present SCALLOP: SCALable isogeny action based on Oriented supersingular curves with Prime conductor, a new isogeny-based group action. Following a standard approach used in CSIDH and OSIDH [CLM+18, CK19], 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. Compared to CSIDH, this is the main benefit of our construction as this data is required to uniquely represent — and efficiently act by — arbitrary group elements, which in turn is a requirement in, e.g., the CSI-FiSh signature scheme by Beullens, Kleinjung and Vercauteren [BKV19].

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 this

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 [EHL+18]) 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 6.5.1), 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 security parameters twice as large as CSI-FiSh.

We provide an implementation of our new group action. The 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.

6.1.1 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 [CLM+18]. 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 $G \times X \to X$ 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 $g \cdot x$. These, and other requirements, have been formalised under the names of Hard Homogenous Space (HHS) [Cou06] or Effective Group Action (EGA) [ADMP20].

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 $\mathfrak{a}^e \cdot E$ 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 $\mathfrak{l}_i \cdot E$ 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 [ADMP20].

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 [SE94]); although, depending on the approximation factor, this step may even have exponential complexity, it is the fastest one in practice.

In this chapter, we describe a method which changes 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, remains 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 $\mathfrak{a}^e \cdot E$, 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 $\left(\prod \mathfrak{l}_i^{e_i}\right) \cdot E$.

In SCALLOP the first step is identical to CSI-FiSh: We use Babai's nearest plane algorithm [Bab86] 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 stabilised by ω, to compute $\mathfrak{l}_i \cdot E$ it is thus sufficient to evaluate ω on $E[\ell_i]$ and determine its eigenspaces.

In Section 6.5.1, 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 6.5.2.

Organisation of the chapter.

The rest of this chapter is organised as follows. Section 6.2 introduces the necessary terminology for oriented supersingular curves used in this chapter. In Section 6.3, we introduce our generic framework for effective orientation and group action computation. Then, we introduce the security notions related to group actions in Section 6.4. In Section 6.5, we explain in detail how the SCALLOP group action works. In Section 6.6, we discuss the concrete instantiation of the scheme. Finally, we analyse one particular angle of attack against the scheme in Section 6.7.

6.2 Orientations of supersingular curves

For the rest of this chapter, we fix a quadratic imaginary field K and a quadratic order $\mathfrak{O}$ of discriminant $D < 0$ in K. In the following, we will only consider *primitive*

$\mathfrak{O}$-orientations of supersingular elliptic curves as defined in Definition 2.2.26 and may omit the word primitive. If (E, ι) is a primitive $\mathfrak{O}$-oriented curve, we call E a $\mathfrak{O}$-*orientable* curve. Further, we consider the following set of (primitively) $\mathfrak{O}$-oriented curves.

Definition 6.2.1. Let $\mathcal{S}_{\mathfrak{O}}(p)$ denote the set of supersingular $\mathfrak{O}$-oriented curves (E, ι) up to isomorphisms and Galois conjugacy.

Note that we consider the elements up to Galois conjugacy. The Frobenius π creates two orientations from each optimal embedding of $\mathfrak{O}$ in a maximal quaternion order of $B_{p,\infty}$, i.e. one on E and one on the twist $E^{(p)}$. This convention diverges from the one taken in [Onu21, Wes22a], where orientations are not considered up to Galois conjugacy. Further, we denote the *class number* of $\mathfrak{O}$, i.e. the size of the class group $\mathrm{Cl}(\mathfrak{O})$, by $h(\mathfrak{O})$.

The following proposition follows from the results proven by Onuki [Onu21, Prop. 3.2, Prop. 3.3, Thm. 3.4] and gives a way to compute $\#\mathcal{S}_{\mathfrak{O}}(p)$.

Proposition 6.2.2. *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 [ACL+22] that

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

Recall that 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 $\mathfrak{a} \cdot (E, \iota) = (E_{\mathfrak{a}}, \iota_{\mathfrak{a}})$. Principal ideals act trivially, thus the operation defines a group action of $\mathrm{Cl}(\mathfrak{O})$ on $\mathcal{S}_{\mathfrak{O}}(p)$, which we also denote by $\cdot$. Onuki proved that this group action is free and transitive in the case of Proposition 6.2.2, see also Section 2.2.5.

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

$$\varphi_{\mathfrak{a}}^E : E \to E_{\mathfrak{a}} = E/E[\mathfrak{a}] \quad \text{and} \quad \iota_{\mathfrak{a}}^E(x) = \frac{1}{n(\mathfrak{a})}\varphi_{\mathfrak{a}}^E \circ \iota(x) \circ \hat{\varphi}_{\mathfrak{a}}^E. \tag{6.1}$$

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 said to be *cyclic*, i.e. it has a cyclic kernel.

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 $\alpha + k$ for $k \in \mathbb{Z}$ will also be a generator.

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 6.3.2.

6.3 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 6.5.2, 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 6.5.1 for an efficient instantiation.

We now present a few generic properties, later in Section 6.3.2, we describe how the orientation by such an order can be made effective.

6.3.1 Factorisation of ideals and decomposition of isogenies

We recall from Eq. (6.1) that if (E, ι_E) is an oriented curve and $\mathfrak{a}$ is an ideal, the action $\mathfrak{a} \cdot (E, \iota_E)$ is computed via an isogeny denoted by $\varphi_{\mathfrak{a}}^E$.

Proposition 6.3.1. *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 6.3.2 below. □

Lemma 6.3.2. *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 $\varphi_{\mathfrak{ab}}^E$ 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 $\mathfrak{b} = \bar{\mathfrak{a}}$ and $\mathfrak{b} \cdot \mathfrak{a} \cdot (E, \iota_E) = (E, \iota_E)$. It follows that $E_{\mathfrak{a}}[\mathfrak{b}] = E_{\mathfrak{a}}[\bar{\mathfrak{a}}] = \ker \hat{\varphi}_{\mathfrak{a}}^E = \varphi_{\mathfrak{a}}^E(E[a]) = \varphi_{\mathfrak{a}}^E(E[\mathfrak{a}] \oplus E[\mathfrak{b}]) = \varphi_{\mathfrak{a}}^E(E[\mathfrak{b}])$.

Lastly, suppose generally that $\gcd(a, b) = c$, writing $a = ca'$, $b = cb'$, $\mathfrak{a} = \mathfrak{c}\mathfrak{a}'$ and $\mathfrak{b} = \bar{\mathfrak{c}}\mathfrak{b}'$. Then $E_{\mathfrak{a}}[\mathfrak{b}] = E_{\mathfrak{a}}[\bar{\mathfrak{c}}] \oplus E_{\mathfrak{a}}[\mathfrak{b}']$. Combining the first case and the second one, we have $E_{\mathfrak{a}}[\bar{\mathfrak{c}}] = \varphi_{\mathfrak{c}}^{E_{\mathfrak{a}'}}(E_{\mathfrak{a}'}[\bar{\mathfrak{c}}]) = \varphi_{\mathfrak{c}}^{E_{\mathfrak{a}'}} \circ \varphi_{\mathfrak{a}'}^E(E[\bar{\mathfrak{c}}]) = \varphi_{\mathfrak{a}}^E(E[\bar{\mathfrak{c}}])$ 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 6.3.2, we will in general specify the tuple $(E, \mathfrak{a}, \mathfrak{b})$ at hand.

6.3.2 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 6.3.1 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 [dQKL+21, DLW22]).

Formally, we have the following definition.

Definition 6.3.3. 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 kernel representations of the isogenies $\varphi_{\mathfrak{L}_1\mathfrak{L}_2}^E$ and $\varphi_{\mathfrak{L}_1}^E$ of degree L_1L_2 and L_1 respectively.

Remark 6.3.4. 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 oriented curve that acts as the shared key, 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 6.3.6, we may directly assume ω_E is represented as (R_E, S_E, Q_E), for simplicity.

6.3.3 Computation of the group action 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 $\mathfrak{a} \cdot (E, \iota_E)$ from the effective orientation ω_E.

By Proposition 6.3.1, 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 $(E_\mathfrak{l}, \iota_{E_\mathfrak{l}}) = \mathfrak{l} \cdot (E, \iota_E)$, from $\mathfrak{l}$ and ω_E, then we can recursively compute the action of any ideal $\mathfrak{a}$ from its factorisation as a product of prime ideals. Therefore, we focus on the two operations of 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 [Vél71]. Thus, the main operation is the computation of $E[\mathfrak{l}]$ from ω_E. Proposition 6.3.5 provides this operation.

Proposition 6.3.5. *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 $\varphi_\mathfrak{l}^E$. □

To computate a generator of $\ker \varphi_\mathfrak{l}^E$ from Proposition 6.3.5 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 6.3.1 to $\omega_E = \hat{\varphi}^E_{\mathfrak{L}_1^{-1}} \circ \varphi^E_{\mathfrak{L}_1\mathfrak{L}_2}$ shows that $\omega_{E_\mathfrak{l}} = [\varphi_\mathfrak{l}^E]_* \omega_E$. 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 $\varphi_\mathfrak{l}^E$ 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 6.5.2.

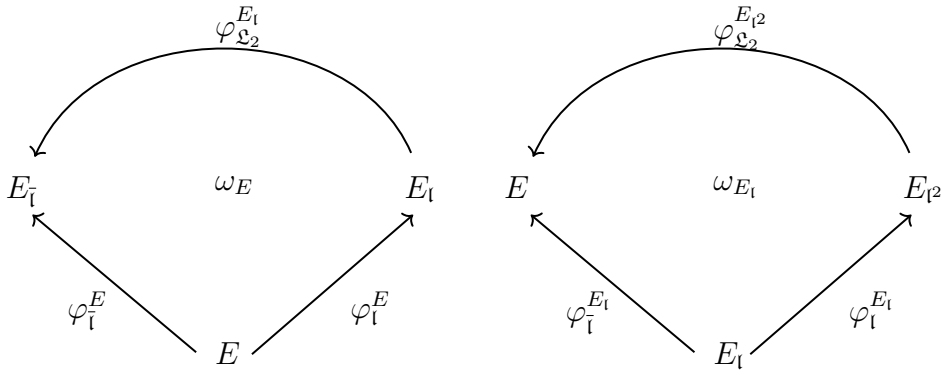

Fig. 6.1 A picture of the effective orientation computation from Proposition 6.3.6.

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 6.3.6. *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 $\hat{\varphi}_{\bar{\mathfrak{l}}}^{E} \circ \varphi_{\mathfrak{L}_2}^{E_\mathfrak{l}} \circ \varphi_{\mathfrak{l}}^{E}$. 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:*

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

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 $\hat{\varphi}_{\bar{\mathfrak{l}}}^{E} \circ \varphi_{\mathfrak{L}_2}^{E_\mathfrak{l}} \circ \varphi_{\mathfrak{l}}^{E}$ for ω_E and $\hat{\varphi}_{\bar{\mathfrak{l}}}^{E_\mathfrak{l}} \circ \varphi_{\mathfrak{L}_2}^{E_{\mathfrak{l}^2}} \circ \varphi_{\mathfrak{l}}^{E_\mathfrak{l}}$ for $\omega_{E_\mathfrak{l}}$ from the factorisation $\mathfrak{O}\alpha = \mathfrak{l}^2\mathfrak{L}_2$. The rest of the proposition follows by applying Lemma 6.3.2 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 6.3.6 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 6.5 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 6.5, 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 [CLM+18], 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 $\varphi_{\mathfrak{l}}^E$ can be directly obtained as one of the two subgroups of order ℓ stable under Frobenius.

6.4 Security of a group action

In this section, we review the best known attacks on the problems underlying our cryptographic group action. Recall the vectorisation and parallelisation problems, defined in Section 2.3.1, associated to a (free and transitive) cryptographic group action of a group G on a set X. A group action $\cdot : G \times X \to X$ is called a *hard homogenous space* if it can be computed efficiently and the vectorisation and parallelisation problems are hard [Cou06]. We call it a *very hard homogenous space* if additionally the following problem is hard.

Problem 6.4.1 (Decisional Parallelisation)**.** Given $x, y, u, v \in X$, decide whether there exists $g \in G$ such that $y = g \cdot x$ and $v = g \cdot u$.

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 [Wes22a]. For simplicity, assume that the factorisation of $\mathrm{disc}(\mathfrak{O})$ is known, and that it has $O(\log\log|\mathrm{disc}(\mathfrak{O})|)$ distinct prime factors[1], as will be the case of our construction.

The two problems $\mathfrak{O}$-Vectorisation and $\mathfrak{O}$-DiffieHellman are equivalent under quantum reductions (see [GPSV21, MZ22] for reductions that are polynomial in the cost of evaluating the group action, or [Wes22a] 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 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 [Wes22a, Fig. 1]).

Problem 6.4.2 ($\mathfrak{O}$-EndRing)**.** Given an effectively oriented curve $(E, \iota_E) \in \mathcal{S}_{\mathfrak{O}}(p)$, compute a basis of the endomorphism ring $\mathrm{End}(E)$.

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

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 [DG16, EHL+20]), 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 [Wes22a, Prop. 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 [Wes22a, Thm. 5]), or even smooth in certain situations (as discussed in [CK19], or more generally [Wes22a, Cor. 6]). We will protect against such attacks by choosing f a large prime. This is in fact one key difference between OSIDH [CK19] and our construction. In OSIDH [CK19], 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 [DD22] 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 [Wes22a, Prop. 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 [Wes22a, Prop. 4]). It relies on Kuperberg's algorithm [Kup05] 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 [CvD10, Iva07, CM22]. 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 [CHVW22] (a generalisation of [CSV22]) 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.

6.5 SCALLOP: a secure and efficient group action

We finally propose an efficient instantiation of the effective group action outlined in Section 6.3. Our main algorithm is given in Section 6.5.2, but we need to motivate our parameter choices first. This is what we do in Section 6.5.1, where we also explain all the required precomputations.

6.5.1 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 6.1.1 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 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

$$\mathfrak{O} = \mathbb{Z} + f\mathfrak{O}_0, \tag{6.2}$$

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 factorisation of f. In practice, we propose to take $\mathfrak{O}_0$ of class number one (e.g. the Gaussian integers) and f a prime number (also for security, as discussed in Section 6.4).

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, is described in the preprint version of the paper underlying this chapter [DFK+23b, Appx. A].

Proposition 6.5.1. *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 generalise 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 6.3, 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 6.6.1, 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 [HW+79, Ch. 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 6.5.2. 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 6.3.1, 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 [BKV19].

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,[2] 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 $\mathfrak{O}$. [BKV19] 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.

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

This explains why we search for f such that $f - (\frac{d_0}{f})$ is as smooth as possible (recall Proposition 6.5.1). 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 6.6).

After having computed a basis for $\mathcal{L}$, the second step, which we do identically to [BKV19], 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 [LN20, Thm. 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 [BKV19].

Finally, any time we are given an ideal class, say $\mathfrak{l}_1^e$, we use Babai's nearest plane algorithm [Bab86] to find a vector $\mathbf{v}$ close to $\mathbf{e} = (e, 0, \ldots, 0)$, 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 6.3, will be proportional to the norm of $\mathbf{e}-\mathbf{v}$. 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 maximises the efficiency of evaluating the group action. We have two requirements: that the effective orientations (E, P_E, Q_E) (see Definition 6.3.3) 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_1L_2 and Q_E has order L_1, hence it is sufficient to choose $L_1L_2 \mid (p^2-1)$.

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 $L = L_1L_2L_3 = \prod_{i=1}^n \ell_i$, 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 $L_1L_2L_3$ 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 6.4). 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 6.3, 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 6.1).

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 [KLPT14]. 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 [DLW22, Alg. 1]. Moreover, we can evaluate any endomorphism of $\mathrm{End}(E_0)$ efficiently, because we have the nice representation made explicit at Example 2.2.25. By a result from [LB20], 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 6.5.3.

Proposition 6.5.3. *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 [Ler22, Alg. 2] (or [FKMT22] described in Section 5.3.1) 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 [DKL+20]. Here, we can even use the alternative key generation method described in [DKL+20,

Appx. D] for better efficiency. Indeed, we can choose almost any isogeny of degree f (by Proposition 6.5.3, 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 6.1. 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 6.3.3.

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

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 6.5.3, E is an $\mathfrak{O}$-orientable curve

Algorithm 6.1: 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 Set $E = E_i$, and $s_E = s_i$.
11 **return** s_E.

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 6.5.3, 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 4, 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 CheckTrace algorithm introduced in [Ler22]. This proves the result. □

6.5.2 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 GroupActionSmall (Algorithm 6.2) that performs the group action computation for one ideal of degree dividing L. The final algorithm GroupAction (described as Algorithm 6.3) is simply the consecutive execution of this subalgorithm 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 6.2 (GroupActionSmall) to compute the action of $\mathfrak{a}$ on (E, s_E).

Fig. 6.2 provides a visualisation of the different isogenies involved in Algorithm 6.2.

Proposition 6.5.5. 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 $E[\mathfrak{L}_1^{-1}] = \langle Q_E\rangle$.

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 Steps 2, 4, 6 and 8 respectively, and their evaluation on the respective $n(\mathfrak{c}_2\mathfrak{a}_3)$ torsion groups in Steps 3, 5,

Algorithm 6.2: 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 $(E_\mathfrak{a}, \iota_{E_\mathfrak{a}})$.

Parse s_E as E, P_E, Q_E.

Compute $\varphi^E_{\mathfrak{b}_1\mathfrak{b}_2}$ from its kernel $\langle \left[\frac{L_1L_2}{n(\mathfrak{b}_1\mathfrak{b}_2)}\right] P_E \rangle$.

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 $\varphi^E_{\mathfrak{b}_1\mathfrak{b}_2}(E[n(\mathfrak{c}_2)L_3])$.

Compute $\varphi^{E_{\mathfrak{b}_1\mathfrak{b}_2}}_{\mathfrak{K}_1\mathfrak{K}_2}$ from its kernel $\langle P^*_{E_{\mathfrak{b}_1\mathfrak{b}_2}} \rangle$.

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 $\varphi^{E_{\mathfrak{b}_1\mathfrak{b}_2}}_{\mathfrak{K}_1\mathfrak{K}_2}(E_{\mathfrak{b}_1\mathfrak{b}_2}[n(\mathfrak{b}_1\mathfrak{b}_2\mathfrak{c}_2)L_3])$.

Compute $\varphi^E_{\mathfrak{c}_1}$ from its kernel $\langle [\frac{L_1}{n(\mathfrak{c}_1)}] Q_E \rangle$

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 $\varphi^E_{\mathfrak{c}_1}(E[n(\mathfrak{c}_2)L_3])$.

Compute $\varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{J}_1}$ from its kernel $\langle Q^*_{E_{\mathfrak{c}_1}} \rangle$

Compute $P_{E_{\mathfrak{L}_1\mathfrak{L}_2}} = \varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{J}_1}(P_{E_{\mathfrak{c}_1}})$ and $\varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{J}_1}(E[n(\mathfrak{c}_1\mathfrak{c}_2)L_3])$.

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 $P_{E_{\mathfrak{b}_1\mathfrak{b}_2}}$.

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 $Q_{E_{\mathfrak{c}_1}}$.

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 $E_{\mathfrak{b}_1\mathfrak{b}_2}[\mathfrak{c}_2\mathfrak{a}_3]$.

Compute $\varphi^{E_{\mathfrak{b}_1\mathfrak{b}_2}}_{\mathfrak{c}_1}$ from its kernel $\langle \left[\frac{L_1}{n(\mathfrak{c}_1)}\right] Q_{E_{\mathfrak{b}_1\mathfrak{b}_2}} \rangle$

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

Compute $\varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{b}_1\mathfrak{b}_2}$ from its kernel $\langle \left[\frac{L_1L_2}{n(\mathfrak{b}_1\mathfrak{b}_2)}\right] P_{E_{\mathfrak{c}_1}} \rangle$

Compute $Q_{E_{\mathfrak{a}_1\mathfrak{b}_2}} = \varphi^{E_{\mathfrak{c}_1}}_{\mathfrak{b}_1\mathfrak{b}_2}(Q_{E_{\mathfrak{c}_1}})$.

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 $E_{\mathfrak{a}_1\mathfrak{b}_2}[\mathfrak{c}_2\mathfrak{a}_3]$.

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 $Q_{E_\mathfrak{a}} = \varphi^{E_{\mathfrak{a}_1\mathfrak{b}_2}}_{\mathfrak{c}_2\mathfrak{a}_3}(Q_{E_{\mathfrak{a}_1\mathfrak{b}_2}})$.

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 6.3.4).

return $s_{E_\mathfrak{a}}$.

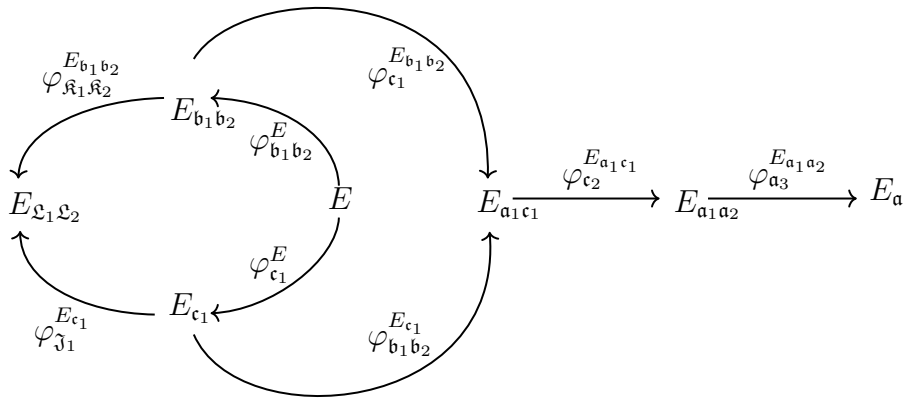

Fig. 6.2 A picture of the isogenies and curves involved in GroupActionSmall.

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 6.3.2 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 6.3.6. 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 6.3.2 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

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

as computed in Step 16,

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

as computed in Step 14 and

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

In Steps 17 and 18, we compute $\varphi_{\mathfrak{c}_2\mathfrak{a}_3}^{E_{\mathfrak{a}_1\mathfrak{b}_2}}$ and applying Lemma 6.3.2 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 6.2 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 6.2, ignoring logarithmic factors, is $\tilde{O}(B)$. □

The full algorithm. Now, Algorithm 6.3 describes the group action evaluation. It is simply made of consecutive executions of GroupActionSmall preceded with a little initialisation.

6.6 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. For a sketch of the signature scheme, we instead refer the reader to the sketch of CSI-FiSh in the preliminaries or to [BKV19] for the detailed description of the scheme.

The security of the new signature scheme based on the SCALLOP group action relies on the problems introduced in Section 6.4. For the concrete instantiation we target two levels of security: matching the security of CSIDH-512 and the one of CSIDH-1024. To

Algorithm 6.3: 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 ≠ 0 do
    a = 1
    for i ∈ {1, ··· , n} do
        if e_i < 0 then
            a = a * l_i^{-1}, e_i = e_i + 1
        else if e_i > 0 then
            a = a * l_i, e_i = e_i − 1
    s_E = GroupActionSmall(s_E, a)
return s_E.
```

obtain class groups of the same size, we take prime conductors of size 256 and 512 bits respectively.

6.6.1 Parameter selection

As outlined in Section 6.5, 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 Section 6.6.2.

In both cases, the computation ran in minutes on a laptop.

6.6.2 Concrete 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

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

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

For these values, 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

$$\begin{aligned}A = &\,609713185630972010670935559853144224720117201550156339729329469291386152543824305586404027203286371218559576094219612254895492407385077835353293836473582216156i\\ &+615420305026332729418500746857711682502001687969362345023251556257118439903978074239505844517514411499242170968625179487146013956635387122252006310139961458627\\ X_P = &\,649662766955987262712642653495434186656775206288198442731398218601213376093486820532674039776128660446087099772024897196330853955057987291522459749571388743645i\\ &+305634628625616960782265852759568451995224409571691436301588240149740241903123884067062444498174462757100863123084027477649307746921497178333469289043565995389\\ Z_P = &\,189973908983857196032146564401173801303257676168549492806318181104067177233687189608217830702585172105613613017410280001889146092837965464078363891392732456735i\\ &+147536290345474169432241022277647529485993368751414461514247010168254478585270137173513005319725222647901855981181744490167495782952306920288856320160951801510\\ X_Q = &\,246981879247557545763993087237034139044900367703281305381071612145231894926493935785318433362583747698473264146540339779183840067932009812209007329466955538i\\ &+467369244782891495439499538266339319885587052010406129668243563967862726424360828941690624656664550571842480155725302787142201839669528155442075391877673039043\\ Z_Q = &\,248137706349969050623203728715630356027266346888181717081745874717163517833815372933044705906891680845109908810627498064202940421960548161243298726554555947i\\ &+538794422715739470837571010481586030525108148491049842442923442344220234238085072105701529147575802448514849470103019165742967961778570505788500695645374180032\end{aligned}$$

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

$$\begin{aligned}\alpha = &-17327891712879992488658400143716156217811012804367932737234052105263563479466035576147102653034852295864721329888440038367531014685905726927026762271760075 8\\ &+ 11106729471624308197513093721752837288547702001147817859082595874813705475944376083267645398956690352890160160287067170471444843426527781965811724438800367 9i.\end{aligned}$$

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 = &1472759983826457766650084250325497270152614398387451020873567932994602207568162203688423845772132160199000373457269445389371361539465703279016172012023515250943437740735287615007324412236 2i\\ &+ 11907705497225532296039026759968931891416410345799613702006580233687054604762529068088420526006041282245016844816620762246411098405345507104748871500047123475604126711073313864874934324924\\ X_P = &4925554444316452034743445647425275931390431368858012377016369363249512321072358408124769514841143750007578202805786829603451881581246711790201217844297691962656275151248012386830047563858 2i\\ &+ 16072832076090261910815201754164796539762892033538169097577317401613453738415144326588031354809592876002402060002592757974274253916296964857126492874282292041837267634867777678867995580993 7\\ Z_P = &5088568723487857101591528146243632397368525211425907794489418877120093767370002342556132637018592901373081443010181013296131507986831761382654380340989093350777749769785210774964944914007 2i\\ &+ 65248389778732897956865554008989144307098612262197312139155367361518332069307328144631484690267671155486388917410120795616790444778420027011367209710594277710901122981203914047875599523144\\ X_Q = &8288790445250350064839699147723505455046784996434709459114263301451

$$
\begin{aligned}
&135553669945348939150398280114042420963163094021928448777402648074 37\\
&682905326644349765739826877381563633921892168505886 23i\\
+\,&384115325031866432970792717602295322333784629575954041931889404422 7\\
&418737588769271073784057747785462633156807245225589715656676533190 01\\
&885882149667396804680202719262505433592686912036183 67\\
Z_Q = \,&763250664170994807123138778288169595703675540573944888418539630497 3\\
&670285126096989935224247985278875178529528896965522748454890953230 07\\
&554634675661332290610653900770818966878567941589328 92i\\
+\,&704147114673890304149013239624126870080211320013996633286905168262 6\\
&261316790162896836150288154800439070365635435825289417775133366481 79\\
&684495850964638523995005358325119476222411080717854 64
\end{aligned}
$$

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.

6.6.3 Performance

Size of public keys. Public keys are represented as effective orientations (E, P_E, Q_E) (see Definition 6.3.3), 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_1L_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.[3] 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 optimised, for instance it does not yet use the $\sqrt{\text{élu}}$ algorithm [BDLS20], we do not expect to gain an order of magnitude by implementing all the possible optimisations. Thus, even if our implementation demonstrates feasibility, it seems that the SCALLOP group action is not yet ready for cryptographic applications.

[3] Our code is available at https://github.com/isogeny-scallop/scallop.

6.7 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 a 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 [Ler22]. 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 [Ler22, Sect. 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 [Ler22, Prop. 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 6.7.1. *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 6.7.1. 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 6.7.1, 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 6.7.1 with the Tate pairing instead of the Weil pairing. However, the curves having trace $\pm 2p$, the Tate pairing is alternating (see [Was08, Thm. 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

[ACC+19] Gora Adj, Daniel Cervantes-Vázquez, Jesús-Javier Chi-Domínguez, Alfred Menezes, and Francisco Rodríguez-Henríquez. On the cost of computing isogenies between supersingular elliptic curves. In Carlos Cid and Michael J. Jacobson Jr:, editors, *SAC 2018*, volume 11349 of *LNCS*, pages 322–343. Springer, Heidelberg, August 2019.

[ACL+22] Sarah Arpin, Mingjie Chen, Kristin E. Lauter, Renate Scheidler, Katherine E. Stange, and Ha T. N. Tran. Orienteering with one endomorphism. Cryptology ePrint Archive, Report 2022/098, 2022. https://eprint.iacr.org/2022/098.

[ADDS21] Martin R. Albrecht, Alex Davidson, Amit Deo, and Nigel P. Smart. Round-optimal verifiable oblivious pseudorandom functions from ideal lattices. In Juan Garay, editor, *PKC 2021, Part II*, volume 12711 of *LNCS*, pages 261–289. Springer, Heidelberg, May 2021.

[ADMP20] Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. Cryptographic group actions and applications. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 411–439. Springer, Heidelberg, December 2020.

[AEN19] Yoshinori Aono, Thomas Espitau, and Phong Q. Nguyen. Random lattices: Theory and practice, 2019.

[AJK+16] Reza Azarderakhsh, David Jao, Kassem Kalach, Brian Koziel, and Christopher Leonardi. Key compression for isogeny-based cryptosystems. In *Proceedings of the 3rd ACM International Workshop on ASIA Public-Key Cryptography*, pages 1–10. ACM, 2016.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 99–108, 1996.

[Bab86] László Babai. On Lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986.

[BBD+22] Jeremy Booher, Ross Bowden, Javad Doliskani, Tako Boris Fouotsa, Steven D. Galbraith, Sabrina Kunzweiler, Simon-Philipp Merz, Christophe Petit, Benjamin Smith, Katherine E. Stange, Yan Bo Ti, Christelle Vincent, José Felipe Voloch, Charlotte Weitkämper, and Lukas Zobernig. Failing to hash into

supersingular isogeny graphs. Cryptology ePrint Archive, Report 2022/518, 2022. https://eprint.iacr.org/2022/518.

[BBEL08] Juliana Belding, Reinier Bröker, Andreas Enge, and Kristin Lauter. Computing hilbert class polynomials. In *Algorithmic Number Theory: 8th International Symposium, ANTS-VIII Banff, Canada, May 17-22, 2008 Proceedings 8*, pages 282–295. Springer, 2008.

[BCC+22] Andrea Basso, Giulio Codogni, Deirdre Connolly, Luca De Feo, Tako Boris Fouotsa, Guido Maria Lido, Travis Morrison, Lorenz Panny, Sikhar Patranabis, and Benjamin Wesolowski. Supersingular curves you can trust. Cryptology ePrint Archive, Report 2022/1469, 2022. https://eprint.iacr.org/2022/1469.

[BCP97] Wieb Bosma, John Cannon, and Catherine Playoust. The Magma algebra system I: The user language. *Journal of Symbolic Computation*, 24(3-4):235–265, 1997.

[BDK+22] Ward Beullens, Samuel Dobson, Shuichi Katsumata, Yi-Fu Lai, and Federico Pintore. Group signatures and more from isogenies and lattices: Generic, simple, and efficient. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 95–126. Springer, Heidelberg, May / June 2022.

[BDLS20] Daniel J. Bernstein, Luca De Feo, Antonin Leroux, and Benjamin Smith. Faster computation of isogenies of large prime degree. In *ANTS-XIV-14th Algorithmic Number Theory Symposium*, volume 4, pages 39–55. Mathematical Sciences Publishers, 2020.

[Ber09] Daniel J. Bernstein. Cost analysis of hash collisions: Will quantum computers make SHARCS obsolete. *SHARCS*, 9:105, 2009.

[BHT97] Gilles Brassard, Peter Hoyer, and Alain Tapp. Quantum algorithm for the collision problem. *arXiv preprint quant-ph/9705002*, 1997. https://arxiv.org/abs/quant-ph/9705002.

[Bis12] Gaetan Bisson. Computing endomorphism rings of elliptic curves under the GRH. *Journal of Mathematical Cryptology*, 5(2):101–114, 2012.

[BJS14] Jean-François Biasse, David Jao, and Anirudh Sankar. A quantum algorithm for computing isogenies between supersingular elliptic curves. In Willi Meier and Debdeep Mukhopadhyay, editors, *INDOCRYPT 2014*, volume 8885 of *LNCS*, pages 428–442. Springer, Heidelberg, December 2014.

[BKM+20] Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Charlotte Weitkämper. On adaptive attacks against Jao-Urbanik's isogeny-based protocol. In *Progress in Cryptology-AFRICACRYPT 2020: 12th*

International Conference on Cryptology in Africa, Cairo, Egypt, July 20–22, 2020, Proceedings 12, pages 195–213. Springer, 2020.

[BKM+21] Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Antonio Sanso. Cryptanalysis of an oblivious PRF from supersingular isogenies. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part I*, volume 13090 of *LNCS*, pages 160–184. Springer, Heidelberg, December 2021.

[BKP20] Ward Beullens, Shuichi Katsumata, and Federico Pintore. Calamari and Falafl: Logarithmic (linkable) ring signatures from isogenies and lattices. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 464–492. Springer, Heidelberg, December 2020.

[BKV19] Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 227–247. Springer, Heidelberg, December 2019.

[BKW20] Dan Boneh, Dmitry Kogan, and Katharine Woo. Oblivious pseudorandom functions from isogenies. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 520–550. Springer, Heidelberg, December 2020.

[BL07] Daniel J. Bernstein and Tanja Lange. Faster addition and doubling on elliptic curves. In Kaoru Kurosawa, editor, *ASIACRYPT 2007*, volume 4833 of *LNCS*, pages 29–50. Springer, Heidelberg, December 2007.

[BS11] Gaetan Bisson and Andrew V. Sutherland. Computing the endomorphism ring of an ordinary elliptic curve over a finite field. *Journal of Number Theory*, 131(5):815–831, 2011.

[BS20] Xavier Bonnetain and André Schrottenloher. Quantum security analysis of CSIDH. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 493–522. Springer, Heidelberg, May 2020.

[CD20] Wouter Castryck and Thomas Decru. CSIDH on the surface. In Jintai Ding and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020*, pages 111–129. Springer, Heidelberg, 2020.

[CD22] Wouter Castryck and Thomas Decru. An efficient key recovery attack on SIDH (preliminary version). Cryptology ePrint Archive, Report 2022/975, 2022. https://eprint.iacr.org/2022/975.

[CDV20] Wouter Castryck, Thomas Decru, and Frederik Vercauteren. Radical isogenies. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 493–519. Springer, Heidelberg, December 2020.

[Cha82] David Chaum. Blind signatures for untraceable payments. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *CRYPTO'82*, pages 199–203. Plenum Press, New York, USA, 1982.

[CHM+23] Wouter Castryck, Marc Houben, Simon-Philipp Merz, Marzio Mula, Sam van Buuren, and Frederik Vercauteren. Weak instances of class group action based cryptography via self-pairings. To appear at CRYPTO 2023. Preprint available at https://eprint.iacr.org/2023/549, 2023.

[CHVW22] Wouter Castryck, Marc Houben, Frederik Vercauteren, and Benjamin Wesolowski. On the decisional Diffie-Hellman problem for class group actions on oriented elliptic curves. *Research in Number Theory*, 8, 2022.

[CJL+17] Craig Costello, David Jao, Patrick Longa, Michael Naehrig, Joost Renes, and David Urbanik. Efficient compression of SIDH public keys. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 679–706. Springer, Heidelberg, April / May 2017.

[CJS14] Andrew Childs, David Jao, and Vladimir Soukharev. Constructing elliptic curve isogenies in quantum subexponential time. *Journal of Mathematical Cryptology*, 8(1):1–29, 2014.

[CK19] Leonardo Colò and David Kohel. Orienting supersingular isogeny graphs. *Number-Theoretic Methods in Cryptology 2019*, 2019.

[CLG09] Denis Xavier Charles, Kristin E. Lauter, and Eyal Z. Goren. Cryptographic hash functions from expander graphs. *Journal of Cryptology*, 22(1):93–113, January 2009.

[CLM+18] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part III*, volume 11274 of *LNCS*, pages 395–427. Springer, Heidelberg, December 2018.

[CLN+20] Craig Costello, Patrick Longa, Michael Naehrig, Joost Renes, and Fernando Virdia. Improved classical cryptanalysis of SIKE in practice. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 505–534. Springer, Heidelberg, May 2020.

[CM22] Wouter Castryck and Natan Vander Meeren. Two remarks on the vectorization problem. Cryptology ePrint Archive, Report 2022/1366, 2022. https://eprint.iacr.org/2022/1366.

[Cor08] Giuseppe Cornacchia. Su di un metodo per la risoluzione in numeri interi dell'equazione $\sum_{h=0}^{n} c_h x^{n-h} y^h = p$. *Giornale di Matematiche di Battaglini*, 46:33–90, 1908.

[Cos19] Craig Costello. Supersingular isogeny key exchange for beginners. In Kenneth G. Paterson and Douglas Stebila, editors, *SAC 2019*, volume 11959 of *LNCS*, pages 21–50. Springer, Heidelberg, August 2019.

[Cos20] Craig Costello. B-SIDH: Supersingular isogeny Diffie-Hellman using twisted torsion. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 440–463. Springer, Heidelberg, December 2020.

[Cou06] Jean-Marc Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291, 2006. https://eprint.iacr.org/2006/291.

[CPV20] Wouter Castryck, Lorenz Panny, and Frederik Vercauteren. Rational isogenies from irrational endomorphisms. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 523–548. Springer, Heidelberg, May 2020.

[CS18] Craig Costello and Benjamin Smith. Montgomery curves and their arithmetic - the case of large characteristic fields. *Journal of Cryptographic Engineering*, 8(3):227–240, September 2018.

[CS20] Daniele Cozzo and Nigel P. Smart. Sashimi: Cutting up CSI-FiSh secret keys to produce an actively secure distributed signing protocol. In Jintai Ding and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020*, pages 169–186. Springer, Heidelberg, 2020.

[CSCJR22] Jorge Chávez-Saab, Jesús-Javier Chi-Domínguez, Samuel Jaques, and Francisco Rodríguez-Henríquez. The SQALE of CSIDH: sublinear Vélu quantum-resistant isogeny action with low exponents. *Journal of Cryptographic Engineering*, 12(3):349–368, September 2022.

[CSV22] Wouter Castryck, Jana Sotáková, and Frederik Vercauteren. Breaking the decisional diffie-hellman problem for class group actions using genus theory: Extended version. *Journal of Cryptology*, 35(4):24, October 2022.

[CV90] David Chaum and Hans Van Antwerpen. Undeniable signatures. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 212–216. Springer, Heidelberg, August 1990.

[CvD10] Andrew M. Childs and Wim van Dam. Quantum algorithms for algebraic problems. *Reviews of Modern Physics*, 82(1):1, 2010.

[DD22] Pierrick Dartois and Luca De Feo. On the security of OSIDH. In Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe, editors, *PKC 2022, Part I*, volume 13177 of *LNCS*, pages 52–81. Springer, Heidelberg, March 2022.

[DDF+21] Luca De Feo, Cyprien Delpech de Saint Guilhem, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Christophe Petit, Javier Silva, and Benjamin Wesolowski. Séta: Supersingular encryption from torsion attacks. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part IV*, volume 13093 of *LNCS*, pages 249–278. Springer, Heidelberg, December 2021.

[Deu41] Max Deuring. Die Typen der Multiplikatorenringe elliptischer Funktionenkörper. In *Abhandlungen aus dem mathematischen Seminar der Universität Hamburg*, volume 14, pages 197–272. Springer, 1941.

[DF17] Luca De Feo. Mathematics of isogeny based cryptography. *arXiv preprint arXiv:1711.04062*, 12, 2017. https://arxiv.org/abs/1711.04062.

[DFK+23a] Luca De Feo, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Simon-Philipp Merz, Lorenz Panny, and Benjamin Wesolowski. SCALLOP: Scaling the CSI-FiSh. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 345–375. Springer, Heidelberg, May 2023.

[DFK+23b] Luca De Feo, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Simon-Philipp Merz, Lorenz Panny, and Benjamin Wesolowski. SCALLOP: scaling the CSI-FiSh. Cryptology ePrint Archive, Report 2023/058, 2023. https://eprint.iacr.org/2023/058.

[DG16] Christina Delfs and Steven D. Galbraith. Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$. *Designs, Codes and Cryptography*, 78(2):425–440, 2016.

[DG19] Luca De Feo and Steven D. Galbraith. SeaSign: Compact isogeny signatures from class group actions. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 759–789. Springer, Heidelberg, May 2019.

[DGS+18] Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. Privacy pass: Bypassing internet challenges anonymously. *PoPETs*, 2018(3):164–180, July 2018.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[DKL+20] Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. SQISign: Compact post-quantum signatures from quaternions and isogenies. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part I*, volume 12491 of *LNCS*, pages 64–93. Springer, Heidelberg, December 2020.

[DKS18] Luca De Feo, Jean Kieffer, and Benjamin Smith. Towards practical key exchange from ordinary isogeny graphs. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part III*, volume 11274 of *LNCS*, pages 365–394. Springer, Heidelberg, December 2018.

[DLW22] Luca De Feo, Antonin Leroux, and Benjamin Wesolowski. New algorithms for the deuring correspondence: SQISign twice as fast. Cryptology ePrint Archive, Report 2022/234, 2022. https://eprint.iacr.org/2022/234.

[DM20] Luca De Feo and Michael Meyer. Threshold schemes from isogeny assumptions. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 187–212. Springer, Heidelberg, May 2020.

[DP96] Ivan Damgård and Torben P. Pedersen. New convertible undeniable signature schemes. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 372–386. Springer, Heidelberg, May 1996.

[DPV19] Thomas Decru, Lorenz Panny, and Frederik Vercauteren. Faster SeaSign signatures through improved rejection sampling. In Jintai Ding and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 10th International Conference, PQCrypto 2019*, pages 271–285. Springer, Heidelberg, 2019.

[dQKL+21] Victoria de Quehen, Péter Kutas, Chris Leonardi, Chloe Martindale, Lorenz Panny, Christophe Petit, and Katherine E. Stange. Improved torsion-point attacks on SIDH variants. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part III*, volume 12827 of *LNCS*, pages 432–470, Virtual Event, August 2021. Springer, Heidelberg.

[DRRT18] Daniel Demmler, Peter Rindal, Mike Rosulek, and Ni Trieu. PIR-PSI: scaling private contact discovery. *Proc. Priv. Enhancing Technol.*, 2018(4):159–178, 2018.

[DSW19] Alex Davidson, Nick Sullivan, and Christopher A. Wood. Oblivious Pseudorandom Functions (OPRFs) using Prime-Order Groups. Internet-draft, Internet Engineering Task Force, 2019. Work in Progress.

[ECS+15] Adam Everspaugh, Rahul Chatterjee, Samuel Scott, Ari Juels, and Thomas Ristenpart. The pythia PRF service. In Jaeyeon Jung and Thorsten Holz, editors, *USENIX Security 2015*, pages 547–562. USENIX Association, August 2015.

[EHL+18] Kirsten Eisenträger, Sean Hallgren, Kristin E. Lauter, Travis Morrison, and Christophe Petit. Supersingular isogeny graphs and endomorphism rings: Reductions and solutions. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 329–368. Springer, Heidelberg, April / May 2018.

[EHL+20] Kirsten Eisenträger, Sean Hallgren, Chris Leonardi, Travis Morrison, and Jennifer Park. Computing endomorphism rings of supersingular elliptic curves and connections to path-finding in isogeny graphs. *Open Book Series*, 4(1):215–232, 2020.

[FHKP13] Eduarda S. V. Freire, Dennis Hofheinz, Eike Kiltz, and Kenneth G. Paterson. Non-interactive key exchange. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *PKC 2013*, volume 7778 of *LNCS*, pages 254–271. Springer, Heidelberg, February / March 2013.

[FIPR05] Michael J. Freedman, Yuval Ishai, Benny Pinkas, and Omer Reingold. Keyword search and oblivious pseudorandom functions. In Joe Kilian, editor, *TCC 2005*, volume 3378 of *LNCS*, pages 303–324. Springer, Heidelberg, February 2005.

[FKMT22] Tako Boris Fouotsa, Péter Kutas, Simon-Philipp Merz, and Yan Bo Ti. On the isogeny problem with torsion point information. In Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe, editors, *PKC 2022, Part I*, volume 13177 of *LNCS*, pages 142–161. Springer, Heidelberg, March 2022.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 537–554. Springer, Heidelberg, August 1999.

[Fou22] Tako Boris Fouotsa. SIDH with masked torsion point images. Cryptology ePrint Archive, Report 2022/1054, 2022. https://eprint.iacr.org/2022/1054.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Heidelberg, August 1987.

[Gal99] Steven D. Galbraith. Constructing isogenies between elliptic curves over finite fields. *LMS Journal of Computation and Mathematics*, 2:118–138, 1999.

[Gal12] Steven D. Galbraith. *Mathematics of public key cryptography*. Cambridge University Press, 2012.

[GGMP20] Steven D. Galbraith, Robert Granger, Simon-Philipp Merz, and Christophe Petit. On index calculus algorithms for subfield curves. In Orr Dunkelman, Michael J. Jacobson Jr., and Colin O'Flynn, editors, *SAC 2020*, volume 12804 of *LNCS*, pages 115–138. Springer, Heidelberg, October 2020.

[GHS02] Steven D. Galbraith, Florian Hess, and Nigel P. Smart. Extending the GHS Weil descent attack. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 29–44. Springer, Heidelberg, April / May 2002.

[GPST16] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 63–91. Springer, Heidelberg, December 2016.

[GPSV21] Steven D. Galbraith, Lorenz Panny, Benjamin Smith, and Frederik Vercauteren. Quantum equivalence of the DLP and CDHP for group actions. *Mathematical Cryptology*, 1(1):40–44, 2021.

[HHK17] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the Fujisaki-Okamoto transformation. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 341–371. Springer, Heidelberg, November 2017.

[HM89] James L. Hafner and Kevin S. McCurley. A rigorous subexponential algorithm for computation of class groups. *Journal of the American mathematical society*, 2(4):837–850, 1989.

[HPS11] Guillaume Hanrot, Xavier Pujol, and Damien Stehlé. Algorithms for the shortest and closest lattice vector problems. *IWCC*, 6639:159–190, 2011.

[HW+79] Godfrey Harold Hardy, Edward Maitland Wright, et al. *An introduction to the theory of numbers*. Oxford university press, 1979.

[IJ13] Sorina Ionica and Antoine Joux. Pairing the volcano. *Mathematics of Computation*, 82(281):581–603, 2013.

[Iva07] Gábor Ivanyos. On solving systems of random linear disequations. 2007. https://arxiv.org/abs/0704.2988.

[JAC+17] David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, Vladimir Soukharev, and David Urbanik. SIKE. Technical report, National Institute of Standards and Technology, 2017. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-1-submissions.

[JAC+19] David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, Vladimir Soukharev, David Urbanik, and Geovandro Pereira. SIKE. Technical report, National Institute of Standards and Technology, 2019. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-2-submissions.

[JD11] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011*, pages 19–34. Springer, Heidelberg, November / December 2011.

[JKK14] Stanislaw Jarecki, Aggelos Kiayias, and Hugo Krawczyk. Round-optimal password-protected secret sharing and T-PAKE in the password-only model. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part II*, volume 8874 of *LNCS*, pages 233–253. Springer, Heidelberg, December 2014.

[JKX18] Stanislaw Jarecki, Hugo Krawczyk, and Jiayu Xu. OPAQUE: An asymmetric PAKE protocol secure against pre-computation attacks. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 456–486. Springer, Heidelberg, April / May 2018.

[JL09] Stanislaw Jarecki and Xiaomin Liu. Efficient oblivious pseudorandom function with applications to adaptive OT and secure computation of set intersection. In Omer Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 577–594. Springer, Heidelberg, March 2009.

[JMV09] David Jao, Stephen D. Miller, and Ramarathnam Venkatesan. Expander graphs based on GRH with an application to elliptic curve cryptography. *Journal of Number Theory*, 129(6):1491–1504, 2009.

[JS14] David Jao and Vladimir Soukharev. Isogeny-based quantum-resistant undeniable signatures. In Michele Mosca, editor, *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014*, pages 160–179. Springer, Heidelberg, October 2014.

[JS19] Samuel Jaques and John M. Schanck. Quantum cryptanalysis in the RAM model: Claw-finding attacks on SIKE. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 32–61. Springer, Heidelberg, August 2019.

[Kan97] Ernst Kani. The number of curves of genus two with elliptic differentials. 1997.

[KF08] Kaoru Kurosawa and Jun Furukawa. Universally composable undeniable signature. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M.

Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP 2008, Part II*, volume 5126 of *LNCS*, pages 524–535. Springer, Heidelberg, July 2008.

[KL20] Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography*. CRC press, 2020.

[KLPT14] David Kohel, Kristin Lauter, Christophe Petit, and Jean-Pierre Tignol. On the quaternion ℓ-isogeny path problem. *LMS Journal of Computation and Mathematics*, 17(A):418–432, 2014.

[KM07] Neal Koblitz and Alfred J. Menezes. Another look at "provable security". *Journal of Cryptology*, 20(1):3–37, January 2007.

[KMPW21] Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Charlotte Weitkämper. One-way functions and malleability oracles: Hidden shift attacks on isogeny-based protocols. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 242–271. Springer, Heidelberg, October 2021.

[Koh96] David Kohel. *Endomorphism rings of elliptic curves over finite fields*. PhD thesis, University of California, Berkeley, 1996.

[Kup05] Greg Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM Journal on Computing*, 35(1):170–188, 2005.

[Kup11] Greg Kuperberg. Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *arXiv preprint arXiv:1112.3333*, 2011. https://arxiv.org/abs/1112.3333.

[KV10] Markus Kirschmer and John Voight. Algorithmic enumeration of ideal classes for quaternion orders. *SIAM Journal on Computing*, 39(5):1714–1747, 2010.

[Lan09] Edmund Landau. *Über die Einteilung der positiven ganzen Zahlen in vier Klassen nach der Mindestzahl der zu ihrer additiven Zusammensetzung erforderlichen Quadrate*. 1909.

[LB20] Jonathan Love and Dan Boneh. Supersingular curves with small noninteger endomorphisms. *Open Book Series*, 4(1):7–22, 2020.

[LD21] Yi-Fu Lai and Samuel Dobson. Collusion resistant revocable ring signatures and group signatures from hard homogeneous spaces. Cryptology ePrint Archive, Report 2021/1365, 2021. https://eprint.iacr.org/2021/1365.

[Ler22] Antonin Leroux. A new isogeny representation and applications to cryptography. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022,*

Part II, volume 13792 of *LNCS*, pages 3–35. Springer, Heidelberg, December 2022.

[LLL82] Arjen K. Lenstra, Hendrik Willem Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:515–534, 1982.

[LN20] Jianwei Li and Phong Q. Nguyen. A complete analysis of the BKZ lattice reduction algorithm. Cryptology ePrint Archive, Report 2020/1237, 2020. https://eprint.iacr.org/2020/1237.

[LPA+19] Lucy Li, Bijeeta Pal, Junade Ali, Nick Sullivan, Rahul Chatterjee, and Thomas Ristenpart. Protocols for checking compromised credentials. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 1387–1403. ACM Press, November 2019.

[LR22] Antonin Leroux and Maxime Roméas. Updatable encryption from group actions. Cryptology ePrint Archive, Report 2022/739, 2022. https://eprint.iacr.org/2022/739.

[LWS21] Patrick Longa, Wen Wang, and Jakub Szefer. The cost to break SIKE: A comparative hardware-based analysis with AES and SHA-3. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part III*, volume 12827 of *LNCS*, pages 402–431, Virtual Event, August 2021. Springer, Heidelberg.

[MM22] Luciano Maino and Chloe Martindale. An attack on SIDH with arbitrary starting curve. Cryptology ePrint Archive, Report 2022/1026, 2022. https://eprint.iacr.org/2022/1026.

[MMP20] Simon-Philipp Merz, Romy Minko, and Christophe Petit. Another look at some isogeny hardness assumptions. In Stanislaw Jarecki, editor, *CT-RSA 2020*, volume 12006 of *LNCS*, pages 496–511. Springer, Heidelberg, February 2020.

[Mon87] Peter L. Montgomery. Speeding the pollard and elliptic curve methods of factorization. *Mathematics of computation*, 48(177):243–264, 1987.

[Mor22] Tomoki Moriya. Masked-degree SIDH. Cryptology ePrint Archive, Report 2022/1019, 2022. https://eprint.iacr.org/2022/1019.

[MP19] Simon-Philipp Merz and Christophe Petit. Factoring products of braids via garside normal form. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 646–678. Springer, Heidelberg, April 2019.

[MZ22] Hart Montgomery and Mark Zhandry. Full quantum equivalence of group action DLog and CDH, and more. In Shweta Agrawal and Dongdai Lin,

editors, *Advances in Cryptology – ASIACRYPT 2022*, pages 3–32, Cham, 2022. Springer Nature Switzerland.

[NIS16] NIST. National Institute of Standards and Technology. Submission Requirements and Evaluation Criteria for the Post-Quantum Cryptography Standardization Process. Available at https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf, 2016. Accessed: 2023-04-04.

[NIS22] NIST. National Institute of Standards and Technology. Call for Additional Digital Signature Schemes for the Post-Quantum Cryptography Standardization Process. Available at https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/call-for-proposals-dig-sig-sept-2022.pdf, 2022. Accessed: 2023-04-04.

[NR19] Michael Naehrig and Joost Renes. Dual isogenies and their application to public-key compression for isogeny-based cryptography. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part II*, volume 11922 of *LNCS*, pages 243–272. Springer, Heidelberg, December 2019.

[NS04] Phong Q. Nguyen and Damien Stehlé. Low-dimensional lattice basis reduction revisited. In Duncan A. Buell, editor, *ANTS 2004*, pages 338–357. Springer, 2004.

[Onu21] Hiroshi Onuki. On oriented supersingular elliptic curves. *Finite Fields and Their Applications*, 69:101777, 2021.

[Pan21] Lorenz Panny. Cryptography on isogeny graphs. *Eindhoven: Technische Universiteit Eindhoven*, 2021.

[Pei20] Chris Peikert. He gives C-sieves on the CSIDH. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 463–492. Springer, Heidelberg, May 2020.

[Pet17] Christophe Petit. Faster algorithms for isogeny problems using torsion point images. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part II*, volume 10625 of *LNCS*, pages 330–353. Springer, Heidelberg, December 2017.

[Piz90] Arnold K. Pizer. Ramanujan graphs and hecke operators. *Bulletin of the American Mathematical Society*, 23(1):127–137, 1990.

[PL17] Christophe Petit and Kristin Lauter. Hard and easy problems for supersingular isogeny graphs. Cryptology ePrint Archive, Report 2017/962, 2017. https://eprint.iacr.org/2017/962.

[Ram13] Srinivasa Ramanujan. First letter to G.H. Hardy. 1913.

[Reg04] Oded Regev. A subexponential time algorithm for the dihedral hidden subgroup problem with polynomial space. *arXiv preprint quant-ph:0406151*, 2004. https://arxiv.org/abs/quant-ph/0406151.

[Rob22a] Damien Robert. Breaking SIDH in polynomial time. Cryptology ePrint Archive, Report 2022/1038, 2022. https://eprint.iacr.org/2022/1038.

[Rob22b] Damien Robert. Some applications of higher dimensional isogenies to elliptic curves (overview of results). Cryptology ePrint Archive, Report 2022/1704, 2022. https://eprint.iacr.org/2022/1704.

[RS06] Alexander Rostovtsev and Anton Stolbunov. Public-Key Cryptosystem Based On Isogenies. Cryptology ePrint Archive, Report 2006/145, 2006. https://eprint.iacr.org/2006/145.

[SC18] M Seshadri Srinath and V Chandrasekaran. Isogeny-based Quantum-resistant Undeniable Blind Signature Scheme. *International Journal of Network Security*, 20(1):9–18, 2018.

[Sch85] René Schoof. Elliptic curves over finite fields and the computation of square roots mod p. *Mathematics of computation*, 44(170):483–494, 1985.

[Sch87] René Schoof. Nonsingular plane cubic curves over finite fields. *Journal of combinatorial theory, Series A*, 46(2):183–211, 1987.

[SE94] Claus-Peter Schnorr and Martin Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical programming*, 66:181–199, 1994.

[Sha94] Igor R. Shafarevich. *Basic algebraic geometry*, volume 2. Springer, 1994.

[SHB21] István András Seres, Máté Horváth, and Péter Burcsi. The legendre pseudorandom function as a multivariate quadratic cryptosystem: Security and applications. Cryptology ePrint Archive, Report 2021/182, 2021. https://eprint.iacr.org/2021/182.

[Sho94] Peter W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134. Ieee, 1994.

[Sil09] Joseph H. Silverman. *The arithmetic of elliptic curves*, volume 106. Springer, 2009.

[SKFB21] Nick Sullivan, Hugo Krawczyk, Owen Friel, and Richard Barnes. OPAQUE with TLS 1.3. Internet-draft, Internet Engineering Task Force, 2021. Work in Progress.

[Smi05] Benjamin A. Smith. Explicit endomorphisms and correspondences. 2005.

[Sto12] Anton Stolbunov. Cryptographic schemes based on isogenies, 2012.

[Sut11] Andrew Sutherland. Computing Hilbert class polynomials with the Chinese remainder theorem. *Mathematics of Computation*, 80(273):501–538, 2011.

[Sut13] Andrew Sutherland. Isogeny volcanoes. *The Open Book Series*, 1(1):507–530, 2013.

[Tan09] Seiichiro Tani. Claw finding algorithms using quantum walk. *Theoretical Computer Science*, 410(50):5285–5297, 2009.

[Tat66] John Tate. Endomorphisms of abelian varieties over finite fields. *Inventiones mathematicae*, 2:134–144, 1966.

[Vél71] Jacques Vélu. Isogénies entre courbes elliptiques. *CR Acad. Sci. Paris, Séries A*, 273:305–347, 1971.

[Voi21] John Voight. *Quaternion algebras*. Springer, 2021.

[vW99] Paul C. van Oorschot and Michael J. Wiener. Parallel collision search with cryptanalytic applications. *Journal of Cryptology*, 12(1):1–28, January 1999.

[Was08] Lawrence C. Washington. *Elliptic curves: number theory and cryptography*. Chapman and Hall/CRC, second edition, 2008.

[Wat69] William C. Waterhouse. Abelian varieties over finite fields. In *Annales scientifiques de l'École normale supérieure*, volume 2, pages 521–560, 1969.

[Wes22a] Benjamin Wesolowski. Orientations and the supersingular endomorphism ring problem. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part III*, volume 13277 of *LNCS*, pages 345–371. Springer, Heidelberg, May / June 2022.

[Wes22b] Benjamin Wesolowski. The supersingular isogeny path and endomorphism ring problems are equivalent. In *62nd FOCS*, pages 1100–1111. IEEE Computer Society Press, February 2022.

[Wes22c] Benjamin Wesolowski. Understanding and improving the Castryck-Decru attack on SIDH, 2022.

[ZSP+18] Gustavo Zanon, Marcos A. Simplício Jr., Geovandro C. C. F. Pereira, Javad Doliskani, and Paulo S. L. M. Barreto. Faster isogeny-based compressed key agreement. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018*, pages 248–268. Springer, Heidelberg, 2018.