Key Exchange and Zero-Knowledge Proofs from Isogenies and Hyperelliptic Curves

Samuel Alexander Dobson

A thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy in Mathematics
The University of Auckland
2022

Abstract

Cryptography plays a vital role in the modern age of computing and security. Of the many branches of cryptography, we primarily focus on two in this thesis. The first is post-quantum secure key exchange from isogenies. Key exchange protocols are critical for setting up secure communication over the internet. We construct a new initial key agreement protocol to replace the classical extended triple Diffie–Hellman (X3DH) scheme in the Signal Protocol, using Supersingular Isogeny Diffie–Hellman (SIDH) for post-quantum security. As part of this work, we introduce a new model capturing the security properties of Signal X3DH, and within this model prove security of our new scheme (SI-X3DH).

The SI-X3DH protocol requires a way of proving knowledge of SIDH secret keys. This brings us to the second primary focus of this thesis: zero-knowledge proofs. As suggested by the name, such schemes allow a statement to be proved without leaking any information other than the validity of the statement. We propose a zero-knowledge proof protocol that fixes a flaw in the security proof of the De Feo–Jao–Plût identification scheme from 2014. We then propose a second protocol that additionally proves correctness of the torsion points in SIDH public keys. These schemes admit the first secure, non-interactive Proofs of Knowledge for SIDH secret keys.

Still in the line of zero-knowledge proofs, we study a primitive used in various classical constructions: unknown-order groups. Our contributions here are threefold. We study the security of recommended parameter sizes for ideal class groups as groups of unknown order, and show that these do not meet their claimed level of security when accounting for Sutherland's primorial steps algorithm in generic groups. In response, we propose new parameters for various levels of security. Secondly, we give a new method of compressing elements of ideal class groups, requiring only 3/4 of their uncompressed size. Finally, we concretely propose a new method of generating groups of unknown order using Jacobians of genus-3 hyperelliptic curves—including an analysis of their security. We show that Jacobians may provide a more efficient choice for unknown-order groups than ideal class groups of imaginary quadratic fields.

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisor Professor Steven D. Galbraith, for the guidance you provided me, your patience, intuition, and feedback, and your support and good nature. I am very grateful to have had you as my supervisor, and could not have wished for a better one.

Besides my supervisor, there are many others in my academic life that deserve thanks. These include my co-supervisor, Associate Professor Giovanni Russello, my advisor, Dr Jeroen Schillewaert, and my many other teachers at the University of Auckland and Massey University. I appreciate your encouragement and support throughout this journey, and for the guidance and time you provided me, as both teachers and friends. Thank you to my co-authors—especially Benjamin Smith and Luca De Feo—for lending your experience and guidance to my work. Special thanks must also go to my colleagues and friends in the department of mathematics at UoA, including Lukas Zobernig, Yan Bo Ti, Trey Li, Shalini Banerjee, Jason LeGrow, Oxana Novikova, and many others. Your friendship and help have been invaluable as I found my place in Auckland and the academic world, and I have learned so much from you all.

I cannot thank enough my parents, Neil and Helen, and my brother, Matthew, for raising and always being there for me, even after I moved away from home. Your love, guidance, encouragement, and care have been so significant and formative, and I have so much to credit you with. I am immensely blessed, grateful, and proud to have such an incredible family.

Thanks also to Greg and Margot Ross, for welcoming me into your home and lives, and for looking after me so kindly and generously in Auckland. I could not ask for better parents-in-law.

Finally, thank you so much to my wife, Sarah. I cannot overstate how much your love and support has meant to me throughout my PhD and beyond. Thank you for your patience, kindness, care, and affirmation. You are such a blessing on my life.

Soli Deo gloria - Glory to God alone.

Please note: an editor has not been used in the construction of this thesis.

Contents

THE UNIVERSITY OF AUCKLAND
SCHOOL OF GRADUATE STUDIES

Co-Authorship Form

This form is to accompany the submission of any PhD that contains published or unpublished co-authored work. **Please include one copy of this form for each co-authored work**. Completed forms should be included in all copies of your thesis submitted for examination and library deposit (including digital deposit), following your thesis Acknowledgements. Co-authored works may be included in a thesis if the candidate has written all or the majority of the text and had their contribution confirmed by all co-authors as not less than 65%.

Please indicate the chapter/section/pages of this thesis that are extracted from a co-authored work and give the title and publication details or details of submission of the co-authored work.

SIDH Proof of Knowledge

Nature of contribution by PhD candidate	Wrote paper, proofs, algorithms, etc.
Extent of contribution by PhD candidate (%)	70

CO-AUTHORS

Name	Nature of Contribution
Luca De Feo	Ideas, discussion, editing, review/feedback
Steven D. Galbraith	Supervision, ideas, wrote part of paper, feedback/review
Lukas Zobernig	Discussion

Certification by Co-Authors

The undersigned hereby certify that:

- the above statement correctly reflects the nature and extent of the PhD candidate's contribution to this work, and the nature of the contribution of each of the co-authors; and
- that the candidate wrote all or the majority of the text.

Name	Signature	Date
Luca De Feo		17/02/2022
Steven D. Galbraith	SDGalbraith	25/02/2022
Lukas Zobernig	Lukas Zobernig	15/02/2022

THE UNIVERSITY OF AUCKLAND
SCHOOL OF GRADUATE STUDIES

Co-Authorship Form

This form is to accompany the submission of any PhD that contains published or unpublished co-authored work. **Please include one copy of this form for each co-authored work**. Completed forms should be included in all copies of your thesis submitted for examination and library deposit (including digital deposit), following your thesis Acknowledgements. Co-authored works may be included in a thesis if the candidate has written all or the majority of the text and had their contribution confirmed by all co-authors as not less than 65%.

Please indicate the chapter/section/pages of this thesis that are extracted from a co-authored work and give the title and publication details or details of submission of the co-authored work.

Post-Quantum Signal Key Agreement with SIDH

Nature of contribution by PhD candidate	Wrote paper, proofs, etc.
Extent of contribution by PhD candidate (%)	85

CO-AUTHORS

Name	Nature of Contribution
Steven D. Galbraith	Supervision, ideas, feedback/review

Certification by Co-Authors

The undersigned hereby certify that:

- the above statement correctly reflects the nature and extent of the PhD candidate’s contribution to this work, and the nature of the contribution of each of the co-authors; and
- that the candidate wrote all or the majority of the text.

Name	Signature	Date
Steven D. Galbraith	S Galbraith	25/02/2022

THE UNIVERSITY OF AUCKLAND
SCHOOL OF GRADUATE STUDIES

Co-Authorship Form

This form is to accompany the submission of any PhD that contains published or unpublished co-authored work. **Please include one copy of this form for each co-authored work**. Completed forms should be included in all copies of your thesis submitted for examination and library deposit (including digital deposit), following your thesis Acknowledgements. Co-authored works may be included in a thesis if the candidate has written all or the majority of the text and had their contribution confirmed by all co-authors as not less than 65%.

Please indicate the chapter/section/pages of this thesis that are extracted from a co-authored work and give the title and publication details or details of submission of the co-authored work.

On the Degree-Insensitive SI-GDH problem and assumption, in the chapter titled “Cryptographic Hardness Assumptions from Isogenies”

Nature of contribution by PhD candidate	Wrote paper, code, ran simulations, etc.
Extent of contribution by PhD candidate (%)	85

CO-AUTHORS

Name	Nature of Contribution
Steven D. Galbraith	Supervision, ideas, review/feedback

Certification by Co-Authors

The undersigned hereby certify that:

- the above statement correctly reflects the nature and extent of the PhD candidate’s contribution to this work, and the nature of the contribution of each of the co-authors; and
- that the candidate wrote all or the majority of the text.

Name	Signature	Date
Steven D. Galbraith	S Galbraith	25/02/2022

THE UNIVERSITY OF AUCKLAND
SCHOOL OF GRADUATE STUDIES

Co-Authorship Form

This form is to accompany the submission of any PhD that contains published or unpublished co-authored work. **Please include one copy of this form for each co-authored work**. Completed forms should be included in all copies of your thesis submitted for examination and library deposit (including digital deposit), following your thesis Acknowledgements. Co-authored works may be included in a thesis if the candidate has written all or the majority of the text and had their contribution confirmed by all co-authors as not less than 65%.

Please indicate the chapter/section/pages of this thesis that are extracted from a co-authored work and give the title and publication details or details of submission of the co-authored work.

Trustless unknown-order groups

Nature of contribution by PhD candidate	Wrote paper, code, literature review, did computations, etc.
Extent of contribution by PhD candidate (%)	70

CO-AUTHORS

Name	Nature of Contribution
Steven D. Galbraith	Supervision, initial idea, review/feedback
Benjamin Smith	Ideas (including compression), review

Certification by Co-Authors

The undersigned hereby certify that:

- the above statement correctly reflects the nature and extent of the PhD candidate's contribution to this work, and the nature of the contribution of each of the co-authors; and
- that the candidate wrote all or the majority of the text.

Name	Signature	Date
Steven D. Galbraith	SDGalbraith	25/02/2022
Benjamin Smith	BSmith	02/09/2021

Introduction

In 1976, Whitfield Diffie and Martin Hellman [DH76] introduced the world to public-key cryptography via their key exchange protocol—now known simply as the Diffie–Hellman (DH) protocol. This scheme revolutionised internet security, allowing a shared secret to be established by two parties at distance, impervious to eavesdroppers.

The original protocol works with the multiplicative group of integers modulo a prime p. Two participants who wish to partake in the protocol will first agree on this prime as well as g, a primitive root modulo p. Each party i will then choose a **private key** uniformly at random,

$$x_i \leftarrow \mathbb{Z}/p\mathbb{Z},$$

and from there compute their corresponding **public key**,

$$X_i \equiv g^{x_i} \pmod{p}.$$

The claimed infeasibility of computing the private key x_i from public key X_i is known as the **discrete logarithm problem** (DLP), and is one of the most pervasive hardness assumptions in all of public-key cryptography. Under the assumption of its hardness, participants can confidently share their public keys X_i with each other without risking leakage of their secrets—hence the "public" designation.

Suppose Alice and Bob wish to conduct a key exchange. Let Alice's private key be a and public key be A, and let Bob's private and public keys be b and B in the same manner. After exchanging public keys A and B, Alice and Bob will both be able to compute the same secret value K as follows:

$$K = B^a = A^b = g^{ab} \pmod{p}.$$

This introduces a second major cryptographic assumption—the **Diffie–Hellman assumption**—which is that an eavesdropper who learns both A and B, and who knows g and p, cannot feasibly compute K. Consequently, K can be used as a shared secret key known only to Alice and Bob.

Key exchange protocols modelled after this original Diffie–Hellman scheme have become ubiquitous throughout public-key cryptography, and form one of the key backbones of security on the internet. Notably, elliptic curves naturally provide an efficient variant of the DH protocol, known as ECDH—with security based on the hardness of the cryptographic elliptic curve DLP (ECDLP) and elliptic curve Diffie–Hellman (ECDH) assumptions.

Post-quantum cryptography

Peter Shor, in 1994, discovered an algorithm for factoring integers and computing discrete logarithms in polynomial-time using a quantum computer [Sho94, Sho97]. This breakthrough was alarming news for the world of public-key cryptography, putting heavily-relied-upon schemes such as RSA, Diffie–Hellman, and elliptic curve cryptography directly in the firing line. If a sufficiently powerful quantum computer should ever be created, it would signal the end of these schemes' security.

While some parties are sceptical about the feasibility of creating such quantum computers, a potential threat like this cannot be ignored. It is important to allow time for new cryptographic schemes to be studied and tested before they are entrusted with real world security, so research and standardisation needs to be done long before it is required. Thus began a search for new cryptographic schemes and primitives, which could still provide security in a world where powerful quantum computers existed. These new schemes, which boast claims of security even in the presence of quantum-enabled adversaries, are labelled **post-quantum**. For clarity, the schemes and assumptions which are not quantum-resistant are often termed **classical**.

To date, progress on construction of quantum computers is slow but steady. IBM recently announced a 127-qubit quantum processor called Eagle [IBM21], in line with their goal of creating a 1000-qubit device by the end of 2023. The current state of affairs and its impact on the security of the aforementioned algorithms is difficult to determine, however, as there are many different models of quantum computation being researched and trialled, with incomparable claims of progress. Until the smoke clears, cryptographic research needs to prepare for the worst.

With the new goal of quantum-resistant assumptions in mind, some old schemes have resurfaced with new purpose, while other entirely new ideas have been proposed. There are five frontrunners in the post-quantum search:

- Lattice-based cryptography: These schemes generally rely on high-dimensional lattices, in which certain problems are easy when given a "nice" (short, fairly orthogonal) basis, but hard when the basis provided consists of large, highly non-orthogonal vectors. Well-studied hard problems include finding the closest point in a lattice to a non-lattice point, and finding short vector(s) in a lattice. The seminal work in this area was by Ajtai in 1996 [Ajt96], while important lattice-based schemes include NTRU [HPS98] for encryption and CRYSTALS-Dilithium [DKL$^+$18] for signatures—both of which (among other lattice-based schemes) are third-round candidates in the NIST standardisation competition [AAA$^+$20].

- Hash-based cryptography: The first work in this area was by Lamport [Lam79], who constructed a one-time signature scheme from hash functions. Merkle [Mer79] improved upon this to construct a finite-use signature scheme using what he called "tree authentication", which we now call a Merkle tree. This area of cryptography has undergone revival due to lack of known quantum attacks, and has also found use in other areas such as zero-knowledge proofs [BBHR18] in recent years.

- Multivariate cryptography: These schemes are based on the hardness of solving systems of polynomial equations in multiple variables, a task which has been proven to be NP-complete [GJ79]. With the ideas first introduced by Matsumoto and Imai [MI88] (and broken seven years later by Patarin [Pat95]), subsequent signature schemes include the Unbalanced Oil and Vinegar scheme [KPG99] and the Rainbow scheme [DCP$^+$19] based on it—another third-round candidate in the NIST standardisation competition. A practical attack on Rainbow was recently presented by Beullens [Beu22].

- Code-based cryptography: While error-correcting codes are an interesting area of research in themselves, they have also been used to construct public-key cryptosystems. Originally introduced by McEliece in 1978 [McE78], such schemes did not attract much interest when first proposed, but have become popular for their conjectured post-quantum security. The hardness of these schemes is based on the (NP-complete) problem of decoding general linear codes. The NIST post-quantum competition includes the McEliece scheme as a code-based contender.
- Isogeny-based cryptography: Discovered (but unpublished) by Couveignes in 1997 [Cou06], isogeny-based cryptography was rediscovered and introduced as a post-quantum primitive by Rostovtsev and Stolbunov in 2006 [RS06]. It gained popularity after the proposal of SIDH by Jao and De Feo [JD11], and also yielded CSIDH [CLM+18], another interesting key exchange scheme. The mathematics and use of isogenies in cryptography will be a major feature in this thesis.

Both lattice- and code-based cryptosystems are generally fast—leveraging linear algebra and parallelisation—but suffer from large key sizes. Hash-based and multivariate cryptosystems offer reasonable signature schemes but have so far not resulted in efficient encryption or key exchange protocols. Finally, isogeny-based schemes have relatively high computational complexity, but offer very short key and signature sizes. For example, SQISign [DKL+20] and CSI-FiSh [BKV19] are both relatively practical isogeny-based signature schemes boasting some of the shortest key and signature sizes of all post-quantum schemes.

We will soon be introduced, in Chapter 1, to an isogeny-based variant of the Diffie–Hellman scheme, known as Supersingular Isogeny Diffie–Hellman (SIDH) [JD11, DJP14]. The spotlight remains on SIDH and isogeny-based key exchange until Chapters 4 and 5.

Zero-knowledge proofs

There is great interest in using cryptography (both classical and post-quantum) to prove various statements in secure ways. For example, we may wish to prove knowledge of some secret without revealing any information about it, or prove that a large computation was done correctly without requiring the verifier to repeat the entire process. Schemes for proving a statement while revealing no other information—only whether the proof is valid or not—are known as **zero-knowledge proofs**.

An accessible example of such proofs can be given in terms of graph isomorphisms. Suppose we have two graphs G_0 and G_1, such that a prover, Peggy, knows an isomorphism $\alpha : G_0 \to G_1$ between them, and wants to convince a verifier, Victor, that she does (without revealing anything about the isomorphism to Victor). To do so, Peggy can choose a random third graph H, isomorphic to G_0 via $\beta : G_0 \to H$ (and therefore isomorphic to G_1 via $\beta \circ \alpha^{-1}$), and send this graph H to Victor. Victor will choose a random challenge bit $b \in \{0, 1\}$ and give b to Peggy. Finally, Peggy will respond with

$$\mathsf{resp} := \begin{cases} \beta & \text{if } b = 0, \\ \beta \circ \alpha^{-1} & \text{if } b = 1. \end{cases}$$

Figure 1 shows the relationship between G_0, G_1, and H visually. Regardless of whether the verifier sends challenge 0 or 1, they will only learn an isomorphism $G_b \to H$, and will learn nothing about the isomorphism between G_0 and G_1. But each challenge offers the verifier a 1/2 chance of catching out a dishonest prover—only a prover who knows α can successfully answer both challenges every time the protocol is run. So after t iterations of this protocol between Victor and Peggy, Victor will be convinced with probability $1 - 1/2^t$ that Peggy does indeed know α.

$$\begin{tikzcd}
G_0 \arrow[dd, dashed, "\alpha"'] \arrow[dr, "\beta"] & \\
& H \\
G_1 \arrow[ur, "\beta \circ \alpha^{-1}"'] &
\end{tikzcd}$$

Figure 1: Zero-knowledge proof of a secret graph isomorphism α

Proving knowledge of a secret key without leaking information about it is a standard idea in cryptography. Many signature schemes, for example the Schnorr signature scheme [Sch91], are Proofs of Knowledge of the secret key used in signing. Chapter 4 proposes a similar zero-knowledge Proof of Knowledge (zk-PoK) for SIDH secret keys. The area of zero-knowledge proof research is vast and varied, though, certainly not limited to just proving knowledge of keys. Some constructions require more complicated setups which may include secret trapdoor knowledge—trapdoors which, if used maliciously, would undermine the schemes' security. There is, therefore, general interest in schemes that can be set up in a manner that does not require trusting other participants to behave honestly, and setups that can be verified as correct after-the-fact. This idea is appropriately named **trustless** setup. When this is possible, and a scheme is set up in this way, the scheme can simply be used without fear of being tricked by someone with inside knowledge.

As we will see in Chapter 7, a common requirement for trustless setup is the ability to generate a group whose order is unknown. In many schemes based on this idea, knowledge of the group order would allow proofs to be forged. We study methods for generating groups of unknown order trustlessly, so that no one is able to learn the group order throughout the process of generating it.

Structure of this thesis

Overall, this thesis has two primary focuses. The first is isogeny-based key exchange—we study and propose some novel cryptographic assumptions related to isogeny-based authenticated key exchange protocols, and show how SIDH can be used to securely instantiate a post-quantum Signal initial key agreement protocol.

Chapter 1 gives an introduction to the mathematics and use of supersingular elliptic curves and isogenies in cryptography, and the SIDH protocol. Chapter 2 delves into cryptographic hardness assumptions related to isogeny-based cryptography, and in particular those related to SIDH. This chapter also introduces some novel assumptions that are used later in this thesis. As part of this chapter, Section 2.2 contains work which is published in [DG19]. Chapter 3 discusses the prominent adaptive attacks on SIDH key exchange protocols, and discusses the difficulties involved with public key validation to prevent such attacks. As discussed below, we then introduce a new zero-knowledge Proof of Knowledge for use in validating SIDH public keys. Finally, Chapter 5 studies the Signal extended triple Diffie–Hellman (X3DH) key establishment protocol and its security model, and proposes a new SIDH-based variant of X3DH which we call

SI-X3DH. The material in this chapter has previously been published in [DG21].

The second branch of this thesis is concerned with zero-knowledge proofs. In this respect, Chapter 4 demonstrates a flaw in the soundness proof of the SIDH identification scheme by De Feo, Jao, and Plût [DJP14], and proposes a new, sound, protocol which is a zero-knowledge Proof of Knowledge (zk-PoK) of SIDH secret keys. This admits the first secure non-interactive zk-PoK for this purpose. The material in Chapter 4 has previously been published in [DDGZ21]. We also delve into the realm of zero-knowledge proofs from classical cryptographic assumptions, discussing systems that require groups of unknown order to achieve a trustless setup. Background material is provided in Chapter 6, where we also introduce our new method of compressing class group elements (Section 6.3). In Chapter 7 we then show that previous estimates of the security of trustless unknown-order groups greatly underestimate certain attacks. We thus suggest new cryptographic parameter sizes for these groups, and also a new way of generating such groups using Jacobians of hyperelliptic curves. Chapters 6 and 7 cover work that is included in the publication [DGS21].

Summary of contributions

The main contributions of this thesis are related to the Supersingular Isogeny Diffie–Hellman (SIDH) key exchange protocol. These original contributions include:

- Demonstration of a flaw in the SIDH identification scheme soundness proof and the proposal of a new protocol that is proved to be sound and zero-knowledge.
- The introduction of a new SIDH public key well-formedness verification method in the shape of a non-interactive zero-knowledge Proof of Knowledge (NIZK-PoK), with respect to a non-standard *distributional* definition of zero-knowledge, which proves the correctness of the two torsion points in SIDH public keys.
- An analysis of the security properties of the Signal initial key agreement protocol, known as extended triple Diffie–Hellman (X3DH), and the proposal of a new security model capturing these properties formally.
- The design of a post-quantum variant of the Signal initial key agreement protocol (X3DH) based on SIDH, using a novel variant of the Fujisaki–Okamoto transformation and the above NIZK-PoK of SIDH keys to avoid adaptive attacks. This includes a proof of security in the new model mentioned in the previous point.
- Analysis of a cryptographic assumption in a previous work on authenticated key exchange based on supersingular isogenies, called the degree-insensitive SIDH assumption, and the provision of computational evidence to demonstrate that this assumption is invalid.

Additionally, we study groups of unknown order and their part in cryptography. We have three main contributions in this area:

- A technical discussion on the impact of Sutherland's algorithm on trustless hidden-order group security, and the proposition of new security estimates and parameter sizes with respect to a new definition of security for these groups.
- A new method of compressing ideal class group elements, improving their representation size to 3/4 of their uncompressed length.
- Analysis of the security of using Jacobians of hyperelliptic curves as groups of unknown order, and a proposal of methods for doing so securely, as well as a comparison of their

efficiency against ideal class groups.

Notation and conventions

Throughout this thesis, $\mathbb{F}_q$ will denote a finite field of cardinality q where, by convention, $q = p^r$ for some power r, with p the (prime) characteristic of the field. If k is a field, $\overline{k}$ will denote its algebraic closure. If R is a ring (or, in particular, a field), $R^\times$ will denote its multiplicative group. E will be used to represent an elliptic curve, while P, Q, K will often be elliptic curve points. As a convention, we will use K_ϕ to denote a point that generates the kernel of a cyclic isogeny ϕ. The identity of the elliptic curve group law (the point at infinity) will be represented as $\mathcal{O}_E$, where the subscript will be omitted when the curve is clear from context. Note that $\mathcal{O}$ will also be used for orders and oracles—the meaning will be clear from the context.

We will use a dashed box around a variable or parameter $\boxed{x}$ to indicate that x is optional and may be omitted (or set to null). When X is a finite set, $x \leftarrow X$ will be used to indicate that x is chosen uniformly at random from X. When $\mathcal{A}$ is an algorithm, $x \leftarrow \mathcal{A}$ will denote running the algorithm $\mathcal{A}$ and assigning its output to x. We use $x \parallel y$ to denote concatenation of the two values x and y, and $\oplus$ to denote the exclusive-OR (XOR) of two bit-strings.

We use big-O notation to represent asymptotic algorithm complexity. Recall that an algorithm $\mathcal{A}$ running in time $T(n)$ on an input of size n has complexity $O(g(n))$ if there exists some constant M such that, for all sufficiently large n, $T(n) \leq Mg(x)$. If this inequality holds for *any* positive constant ϵ for sufficiently large $n > N$, then $\mathcal{A}$ has complexity $o(g(x))$. Recall that

$$\widetilde{O}(x) = O((\log x)^c \cdot x)$$

for some constant c, and for sub-exponential algorithms,

$$L_x(\alpha) = \exp\left[(1 + o(1))(\log x)^\alpha (\log\log x)^{1-\alpha}\right]$$

for $0 \leq \alpha \leq 1$. Finally, we let negl be a negligible function, such that $|\mathsf{negl}(\kappa)| < 1/\mathsf{poly}(\kappa)$ for all polynomials poly and sufficiently large κ.

Chapter 1

Isogenies and SIDH

A significant proportion of this thesis relates to the Supersingular Isogeny Diffie–Hellman (SIDH) identification and key exchange protocols. Therefore, it makes sense to begin with an introduction to isogenies, supersingular elliptic curves, and isogeny-based cryptography. Elliptic curves have been a cornerstone of classical cryptography for the past two decades due to the efficiency and small key sizes of many primitives using them. They are well-understood mathematically, offer useful features such as pairings, and cryptographic hardness assumptions such as the elliptic curve discrete logarithm problem (ECDLP) have been thoroughly studied and tested.

Despite the looming threat of Shor's quantum algorithm against classical cryptosystems, it seems a shame to throw away these decades of research around elliptic curves. The field of isogeny-based cryptography as a post-quantum contender is therefore interesting for its use of elliptic curve theory among other reasons. Isogeny-based cryptography also features some of the shortest key sizes of all post-quantum primitive proposals.

This chapter shall serve as a brief introduction to the topic of isogeny-based cryptography, highlighting the Supersingular Isogeny Diffie–Hellman (SIDH) key exchange protocol, which is of particular interest and will be revisited many times throughout this thesis.

1.1 Elliptic curves and isogenies

Excellent references for background and detail on elliptic curves include Silverman's [Sil09] and Galbraith's [Gal12] books—both considered canonical references for these topics. We assume some familiarity with these fundamentals but provide a compact refresher.

Recall that an **algebraic curve** is a dimension one algebraic variety. A non-singular projective algebraic curve of genus one with a distinguished point $\mathcal{O}$ is called an **elliptic curve**. Let E be such a curve.

In all fields k of characteristic $\operatorname{char} k \neq 2, 3$, an elliptic curve can be given by a short Weierstrass equation

$$E \ : \ y^2 = x^3 + Ax + B, \tag{1.1}$$

for coefficients $A, B \in k$ (that is, E is **defined over** k). Non-singularity follows if the discriminant of the curve, $\Delta_E = -16(4A^3 + 27B^2)$, is non-zero. Note that the Weierstrass equation is affine, but we are really identifying a projective curve (the projective closure), on which the point at infinity $\mathcal{O}$ lies.

In this thesis, we are interested only in elliptic curves over finite fields $\mathbb{F}_q$ of cardinality $q = p^n$, where $p \geq 5$ is the characteristic of the field. This will be assumed henceforth.

The j-invariant of an elliptic curve with Weierstrass equation $y^2 = x^3 + Ax + B$ is defined as follows:

Definition 1.1 (j-invariant of an elliptic curve E)**.**

$$j(E) = 1728 \frac{4A^3}{4A^3 + 27B^2}$$

Two elliptic curves E, E' are isomorphic (over the algebraic closure $\overline{\mathbb{F}}_q$) if and only if their j-invariants are equal. This does not mean, however, that the isomorphism is defined over $\mathbb{F}_q$. For example, if d is a quadratic non-residue in $\mathbb{F}_q$, then $dy^2 = x^3 + Ax + B$ defines an elliptic curve E^d with the same j-invariant as E, but to which the isomorphism from E is not defined over F_q. This curve is known as a **quadratic twist** of E, since the isomorphism between E and E^d is only defined over a quadratic extension of $\mathbb{F}_q$.

We can also provide a converse mapping, from j-invariants to elliptic curves. For any $j \in k$, $j \neq 0, 1728$, define the curve

$$E_j \ : \ y^2 = x^3 + \frac{3j}{1728 - j} x + \frac{2j}{1728 - j}. \tag{1.2}$$

Then $j(E_j) = j$. The curves $E_0 \ : \ y^2 = x^3 + 1$ and $E_{1728} \ : \ y^2 = x^3 + x$ have j-invariant 0 and 1728 respectively.

For a field k, $E(k)$ denotes the set of k-rational points (points whose coordinates lie in k) on E:

$$E(k) = \left\{ (x, y) \in k^2 \;\middle|\; x^3 + Ax + B = y^2 \right\} \cup \{\mathcal{O}_E\}, \tag{1.3}$$

where $\mathcal{O}_E$ is the point at infinity on E. It is well known that $E(k)$ forms an abelian group under the "chord and tangent" rule (with identity $\mathcal{O}_E$)—the foundation of classical elliptic curve cryptography—making E an abelian variety. This group structure is inherited from the divisor class group $\mathrm{Pic}^0(E)$ of E's Jacobian variety $J_E \cong E$, but can be easily computed geometrically. The divisor class group and Jacobian are discussed further in Chapter 6.

The number of points in $E(\mathbb{F}_q)$ is bounded by Hasse's Theorem, which states that

$$\#E(\mathbb{F}_q) = q + 1 - t, \tag{1.4}$$

where $|t| \leq 2\sqrt{q}$ is the **trace of Frobenius**. It is a fact that the trace of Frobenius of the quadratic twist of E is $-t$.

An **isogeny** of degree d (or a d-isogeny) is a surjective morphism with a finite kernel. An isogeny $\phi : E \to E'$ preserves the point at infinity, so that $\phi(\mathcal{O}_E) = \mathcal{O}_{E'}$. Isogenies are, therefore, homomorphisms of the group structure on the elliptic curves. If two elliptic curves have an isogeny between them, they are called **isogenous**, and it is a well-known theorem of Tate that two elliptic curves E, E' are isogenous over $\mathbb{F}_q$ if and only if they have the same cardinality $\#E(\mathbb{F}_q) = \#E'(\mathbb{F}_q)$ [Tat66].

Almost all isogenies we will encounter in this thesis are **separable**, which implies that the order of the kernel is equal to the degree of the isogeny. What is more, we shall usually only deal with **cyclic** isogenies—isogenies whose kernel subgroups are cyclic—so that $\ker(\phi) = \langle K \rangle$ for a point K of order d, if ϕ has degree d.

Isogenies $\phi : E \to E$ whose codomain and domain coincide are called **endomorphisms**. All endomorphisms of E, together with the zero-map on E (which sends all points on E to $\mathcal{O}_E$), form a ring under the operations of composition and point-wise addition. This is known as the **endomorphism ring** of E, and is denoted by $\mathrm{End}(E)$.

An example of an isogeny is the multiplication-by-n map, $[n] : E \to E$. This isogeny is separable when n is coprime to p, and has degree n^2. Such multiplication maps are contained in the endomorphism ring $\mathrm{End}(E)$ for all elliptic curves E. As the name suggests, the multiplication-by-n map sends points $P \mapsto [n]P$ on E. These isogenies $[n]$ can be considered an embedding of $\mathbb{Z}$ into (the centre of) $\mathrm{End}(E)$. Over fields of positive characteristic, including all finite fields we work with in this thesis, $\mathrm{End}(E)$ is bigger than just $\mathbb{Z}$, though. We say that such curves with endomorphism rings larger than $\mathbb{Z}$ have **complex multiplication**. For example, the (q-power) Frobenius endomorphism is defined as

$$\pi_q((x, y)) = (x^q, y^q). \tag{1.5}$$

This is an example of a purely inseparable isogeny, which every E over $\mathbb{F}_q$ possesses in its endomorphism ring, but is not (in general) in the image of $\mathbb{Z}$.

The kernel of the multiplication-by-n isogeny is, by definition, the set of points $P \in E(\overline{\mathbb{F}}_q)$ of order dividing n. These points form the n-torsion subgroup of E, denoted by $E[n]$. When n is not divisible by p, these torsion subgroups have the form

$$E[n] \cong \frac{\mathbb{Z}}{n\mathbb{Z}} \times \frac{\mathbb{Z}}{n\mathbb{Z}}. \tag{1.6}$$

Of special importance in the theory of elliptic curves, though, are the p^r-torsion subgroups (for all powers r). We have two cases,

$$E[p^r] \cong \begin{cases} \{\mathcal{O}_E\} & \text{or} \\ \mathbb{Z}/p^r\mathbb{Z}. \end{cases} \tag{1.7}$$

In the first case, the curve is called **supersingular**. These curves have especially large, non-commutative endomorphism rings over $\overline{\mathbb{F}}_q$—isomorphic to maximal orders in a four-dimensional quaternion algebra (specifically, the unique quaternion algebra over $\mathbb{Q}$ ramified at exactly p and ∞), with rank four as $\mathbb{Z}$-modules. It is these supersingular curves that are of the most interest to us in this thesis. The trace of Frobenius of a supersingular elliptic curve is equal to 0 modulo p, and the converse also holds. Note, then, that supersingularity is an isogeny invariant, by Tate's isogeny theorem and [Sil09, Exercise 5.13].

Otherwise (the second case), if a curve is not supersingular, it is known as **ordinary**. The endomorphism rings of ordinary curves are isomorphic to dimension two orders inside the ring of integers of $K = \mathbb{Q}\left(\sqrt{t^2 - 4q}\right)$, an imaginary quadratic extension, where t is the trace of Frobenius of the curve.

Every isogeny $\phi : E_1 \to E_2$ has a unique **dual** isogeny, denoted by $\widehat{\phi} : E_2 \to E_1$, of the same degree d [Sil09, Theorem III.6.1]. This dual isogeny has the defining property that $\widehat{\phi} \circ \phi = [d]$. An isogeny of degree 1 is an isomorphism, whose inverse is the dual, since [1] (their composition) is the identity map on E.

Vélu [Vél71] presented explicit formulae for computing a degree-ℓ separable isogeny with a given kernel G on an elliptic curve E. Vélu's algorithm runs in time polynomial in the degree of the isogeny (or equivalently, the size of its kernel). This makes knowledge of an isogeny interchangeable with knowledge of its kernel. Two isogenies are equivalent if their kernels are equal (that is, the isogenies are equal up to post-composition with an isomorphism). We thus have a 1–1 correspondence between finite subgroups of E and separable isogenies $\phi : E \to E'$ up to equivalence [Mum08, pg. 72, Theorem 4].

Every isogeny of composite degree can be factored as the composition of a sequence of isogenies of prime degree [Gal12, Theorem 25.1.2]. This makes the computation of smooth-power isogenies efficient, such as those of degree 2^n and 3^m, which will feature heavily in this thesis.

1.2 The supersingular isogeny graph

The $\mathbb{F}_q$-isogeny class of a curve E is the set of $\mathbb{F}_q$-isomorphism classes of elliptic curves over $\mathbb{F}_q$ that are isogenous over $\mathbb{F}_q$ to E. Note that, because $\mathbb{F}_q$ is not algebraically closed, these isomorphism classes cannot be identified with their j-invariants in general—for example, a supersingular elliptic curve and its quadratic twist will both have the same j-invariant, but will not be isomorphic over $\mathbb{F}_q$, as we have seen above. If we work over the algebraic closure of $\mathbb{F}_q$, this problem goes away, since all isomorphisms will then be defined.

Definition 1.2 (The general isogeny graph)**.** Let ℓ be a prime. The ℓ-isogeny graph over $\mathbb{F}_q$ is the directed graph whose vertices are $\mathbb{F}_q$-isomorphism classes of elliptic curves over $\mathbb{F}_q$, and whose edges correspond to degree-ℓ isogenies (up to equivalence) between the corresponding $\mathbb{F}_q$-isomorphism classes.

Over an algebraically closed field, $\overline{\mathbb{F}}_q$, the ℓ-isogeny graph is $(\ell + 1)$-regular—that is, every vertex has $(\ell + 1)$ unique edges incident on it—corresponding to the $(\ell + 1)$ different degree-ℓ subgroups of $E[\ell]$. Because $\mathbb{F}_q$ is not algebraically closed, however, not all of these isogenies will be defined over $\mathbb{F}_q$.

We care, in particular, about the supersingular curves. Because supersingularity is isogeny-invariant, we can simply consider the connected components of the isogeny graph consisting of the supersingular curves (the "supersingular components") and ignore the ordinary components. The ordinary components are very different from the supersingular components and have found various uses, but are not important for this thesis, so shall not be discussed here.

Definition 1.3 (The supersingular isogeny graph)**.** For a prime ℓ, the supersingular ℓ-isogeny graph over $\mathbb{F}_q$ consists of the $\mathbb{F}_q$-isomorphism classes of supersingular elliptic curves and the ℓ-isogenies between them. It is a subgraph of the degree-ℓ isogeny graph over $\mathbb{F}_q$.

Every supersingular curve E is isomorphic to one defined over $\mathbb{F}_{p^2}$, and $j(E) \in \mathbb{F}_{p^2}$. Therefore, it is common to work over $\mathbb{F}_{p^2}$ when dealing with supersingular curves. The number of $\overline{\mathbb{F}}_p$-isomorphism classes of supersingular curves over $\mathbb{F}_{p^2}$ is $\lfloor p/12 \rfloor + \epsilon_p$ where $\epsilon_p = 0, 1, 1, 2$ for $p \equiv 1, 5, 7, 11 \pmod{12}$ respectively [Gal12, Theorem 9.11.12].

Recall that a path in a graph is a sequence of directed edges, where the start of each edge is the same as the end of the previous. A graph is strongly connected if there is a (directed) path from every vertex to every other vertex.

The supersingular isogeny graph consisting of degree-ℓ isogenies (defined over $\overline{\mathbb{F}}_p$) is strongly connected, for any prime ℓ [Gal12, Theorem 25.3.17]. Graphs of supersingular elliptic curves are Ramanujan [Piz98, CLG09], a particular type of expander graph, implying excellent mixing properties—it is only a short walk from any node to any other node due to the highly-connected nature of the graph.

In the world of supersingular elliptic curves, there are two values of the j-invariant of particular note: $j(E) = 0$ (corresponding to $A = 0$ in the Weierstrass equation 1.1) and $j(E) = 1728$ (corresponding to $B = 0$). Curves with these j-invariants have non-trivial automorphisms (endomorphisms of degree 1) on them, in addition to the usual $[1]$ and $[-1]$ maps. In the first case, there is an automorphism $\eta_6 : (x, y) \mapsto (\zeta_3 x, -y)$ of order six, where ζ_3 is a primitive cube root of unity. In the latter case, we have instead an automorphism of order four, $\eta_4 : (x, y) \mapsto (-x, iy)$, where i is a primitive fourth root of unity.

Consider an isogeny $\phi : E_0 \to E_1$, whose dual $\widehat{\phi}$ has kernel K. If η is an automorphism on E_1 which does not fix K, then $\eta \circ \phi$ is an isogeny equivalent to ϕ, but whose dual is not equivalent to $\widehat{\phi}$. Thus, although there is only one edge ϕ going from E_0 to E_1 (or, more precisely, their isomorphism classes) in the isogeny graph, there will be multiple distinct edges returning from E_1 to E_0, whenever such non-trivial automorphisms exist on E_1. As just discussed, this only happens on curves with j-invariant 0 or 1728. Therefore, away from these two j-invariants, we can consider the supersingular isogeny graph to be undirected and unweighted: every edge has a single “dual” edge with the exception of edges to these curves with extra automorphisms.

1.3 Supersingular Isogeny Diffie–Hellman

One of the most prominent contributions in the sphere of isogeny-based cryptography is the Supersingular Isogeny Diffie–Hellman (SIDH) protocol, introduced by David Jao and Luca De Feo in 2011 [JD11] and later extended with Jérôme Plût [DJP14]. The critical ingredient in this key exchange scheme is the idea of providing the images of torsion bases under the secret isogenies to facilitate a commutative square between the two participants. As we shall discuss later in this thesis, the SIDH protocol has formed the basis of the SIKE [ACC$^+$17] key encapsulation mechanism (KEM) which is an alternative third-round candidate in the NIST post-quantum standardisation competition [AAA$^+$20].

We set the scene for the SIDH protocol with various public parameter choices. We have a prime of the form $p = \ell_1^{e_1}\ell_2^{e_2}f \pm 1$, where ℓ_1, ℓ_2 are distinct small primes, f is an integer cofactor, and $\ell_1^{e_1} \approx \ell_2^{e_2}$. This prime p defines the field $\mathbb{F}_{p^2}$ which we shall work over. Typically, we let $\ell_1 = 2$ and $\ell_2 = 3$, as is the case in SIKE, because computing isogenies of small-prime degree is generally more efficient than using isogenies of larger degree. There are some SIDH variants which use other primes, however, such as eSIDH [COR21].

As an analogue of the base g in the Diffie–Hellman protocol, we fix a supersingular elliptic curve E_0 such that $E_0(\mathbb{F}_{p^2})$ has cardinality $(\ell_1^{e_1}\ell_2^{e_2}f)^2$. This ensures that E_0 has full $\mathbb{F}_{p^2}$-rational $\ell_1^{e_1}$- and $\ell_2^{e_2}$-torsion subgroups, and therefore also that the ℓ_1- and ℓ_2-power isogeny computations are done over $\mathbb{F}_{p^2}$ too for efficiency, rather than requiring larger field extensions.

On E_0, we choose a basis (or generating set) for both the $\ell_1^{e_1}$- and $\ell_2^{e_2}$-torsion subgroups. By Equation 1.6, such bases consist of two independent points of the correct order, so denote by $\{P_i, Q_i\}$ the chosen basis for $\ell_i^{e_i}$ (such that $E_0[\ell_i^{e_i}] = \langle P_i, Q_i \rangle$). Throughout this thesis, we will also refer to the two participants in a key exchange by the standard names Alice and Bob—we accordingly treat both the index 1 and the subscript A as equivalent for Alice, while B and 2 will be used interchangeably for Bob’s information. This is done for clarity in various situations and for consistency with existing literature.

In full, then, SIDH public parameters consist of $\mathsf{pp} = (\ell_1, \ell_2, e_1, e_2, p, E_0, P_1, Q_1, P_2, Q_2)$. Suppose two parties Alice and Bob have agreed on these public parameters and wish to participate in a key exchange. The secret key of each party will be an isogeny with domain E_0. To generate an isogeny of degree $\ell_i^{e_i}$ from E_0, a participant could generate randomly chosen secret integers $a_i, b_i \in \mathbb{Z}/\ell_i^{e_i}\mathbb{Z}$, not both divisible by ℓ_i, and compute the isogeny with kernel $K_i = \langle [a_i]P_i + [b_i]Q_i \rangle$ via Vélu's formulae. However, Galbraith et al. [GPST16, Lemma 2.1] formally present the idea of "equivalent keys", which were implicit in previous works including Costello et al. [CLN16] and are a common theme across other areas of cryptography:

Definition 1.4 (Equivalent SIDH keys)**.** Two SIDH secret keys (a, b) and (a', b') are **equivalent** if they generate the same subgroup for any basis of the $\ell_i^{e_i}$-torsion subgroup. This is true when $(a', b') = (\theta a, \theta b)$ for some $\theta \in (\mathbb{Z}/\ell_i^{e_i}\mathbb{Z})^\times$. Assume that a is not divisible by ℓ_i, then we can choose $\theta \equiv a^{-1} \pmod{\ell_i^{e_i}}$, giving an equivalent key $(1, b')$ with $b' \equiv a^{-1}b \pmod{\ell_i^{e_i}}$. Similarly, if b is not divisible by ℓ_i, we can invert it and obtain equivalent key $(a', 1)$. Hence, we realise a shorter representation of secret keys, without loss of generality, requiring only a single element of $(\mathbb{Z}/\ell_i^{e_i}\mathbb{Z})$ and one extra bit (to determine the case).

From this idea, rather than working with the full generality of choosing two integers a_i, b_i, we restrict to secret keys the form $(1, \alpha)$ for simplicity. This only reduces the size of the effective secret keyspace by a single bit, and is exactly what is done in SIKE and many other protocols. Alice will simply choose a single integer $\alpha \leftarrow \mathbb{Z}/\ell_1^{e_1}\mathbb{Z}$ and compute the isogeny

$$\phi_A : E_0 \to E_A = E_0/\langle P_1 + [\alpha]Q_1 \rangle \tag{1.8}$$

which has degree $\ell_1^{e_1}$. In kind, Bob will sample $\beta \leftarrow \mathbb{Z}/\ell_2^{e_2}\mathbb{Z}$ and compute ϕ_B of degree $\ell_2^{e_2}$ with kernel $\langle P_2 + [\beta]Q_2 \rangle$.

All three of ϕ_A, its kernel $\langle P_1 + [\alpha]Q_1 \rangle$, and the integer α will be referred to interchangeably as Alice's "secret key". Alice's public key takes the form $(E_A, \phi_A(P_2), \phi_A(Q_2))$, providing both the codomain of her secret isogeny and the image of the $\ell_2^{e_2}$-torsion subgroup under said isogeny. The reason for the latter points being included is so that Bob is able to "transport" his secret isogeny to E_A, and permits a kind of commutativity.

When given Alice's public key, Bob will compute the isogeny

$$\phi_{AB} : E_A \to E_{AB} = E_A/\langle \phi_A(P_2) + [\beta]\phi_A(Q_2) \rangle. \tag{1.9}$$

Similarly, given Bob's public key $(E_B, \phi_B(P_1), \phi_B(Q_1))$, Alice will compute

$$\phi_{BA} : E_B \to E_{BA} = E_B/\langle \phi_B(P_1) + [\alpha]\phi_B(Q_1) \rangle. \tag{1.10}$$

Because isogenies are group homomorphisms and these kernels have trivial intersection, the codomains of both these curves will be isomorphic to the curve

$$E_{AB} \cong E_{BA} \cong E_0/\langle P_1 + [\alpha]Q_1, P_2 + [\beta]Q_2 \rangle. \tag{1.11}$$

Consequently, the j-invariants of these curves computed by the participants will be equal, and can be used as a shared secret key. In fact, because Vélu's formulae produce normalised isogenies, both Alice and Bob will actually arrive at exactly the same curve, not just isomorphic ones [Leo20].

$$\begin{tikzcd}
E_0 \arrow[r, "\phi_A"] \arrow[d, "\phi_B"'] & E_A \arrow[d, "\phi_{AB}"] \\
E_B \arrow[r, "\phi_{BA}"'] & E_{AB}
\end{tikzcd}$$

Figure 1.1: Commutative diagram of SIDH, where $\ker(\phi_{BA}) = \phi_B(\ker(\phi_A))$ and $\ker(\phi_{AB}) = \phi_A(\ker(\phi_B))$.

Figure 1.1 depicts the commutative diagram making up the key exchange.

Throughout this thesis, we will use the function $\mathrm{SIDH}_{\mathsf{pp}}(\cdot,\cdot)$ to represent this protocol with respect to public parameters pp, whose output is the final j-invariant derived by both participants. Generally, the public parameters will be clear from context, so they may be omitted for ease of notation. The arguments to $\mathrm{SIDH}_{\mathsf{pp}}$ will be the two public keys of the participants, because clearly the result is independent of which participant computed the value (using their secret key). Specifically, if β is the secret key corresponding to the public key $K_B = (E_B, P_1', Q_1')$, then

$$\mathrm{SIDH}_{\mathsf{pp}}((E_A, P_2', Q_2'), K_B) = j(E_A/\langle P_2' + [\beta]Q_2'\rangle). \tag{1.12}$$

We will discuss the cryptographic hardness assumptions pertaining to isogenies and the SIDH protocol in Chapter 2.

1.4 Some useful lemmas

We collect here some basic definitions and lemmas that we will use repeatedly later in this thesis—especially in Chapter 4. These lemmas include the use of the Weil pairing, which we will introduce in more detail in Section 3.3.

Lemma 1.5. *Let E be an elliptic curve, let $m = \ell^e$ where $\ell \neq p$ is prime and e is a positive integer, and let (P, Q) be a basis of $E[m]$. Let $R = [a]P + [b]Q$ and $S = [c]P + [d]Q$. The following conditions are equivalent:*

(i) (R, S) forms a generating set (or basis) of $E[m]$.

(ii) ℓ does not divide $ad - bc$, or in other words, the matrix

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

is invertible modulo m.

(iii) The value of the m-th Weil paring $w = e_m(R, S)$ has order m. That is, $w^{m/\ell} \neq 1$.

Proof. (i) ⇒ (ii): Both (P, Q) and (R, S) are bases (or generating sets) of the same torsion subgroup $E[m]$. Hence, we must be able to write (P, Q) in terms of (R, S), and we can represent this transformation as a matrix $\tilde{A}$. Composing these shifts between (P, Q) and (R, S) and back, it becomes clear that $A\tilde{A} = I_2$, the identity matrix. So $\tilde{A} = A^{-1}$, and evidently A is invertible, therefore so too is its determinant $ad - bc \pmod{m}$.

(ii) ⇒ (iii): We have that

$$w = e_m(R, S) = e_m([a]P + [b]Q, [c]P + [d]Q).$$

Then, since e_m is bilinear, $w = e_m(P, Q)^{ad-bc}$. Now $e_m(P, Q)$ has order m because e_m is surjective onto the group of m-th roots of unity (c.f. [Sil09, Corollary III.8.1.1]), and since ℓ does not divide $ad - bc$, then w must also have exact order $m = \ell^e$.

(iii) ⇒ (i): Recall from Equation 1.6 that $E[m] \simeq \mathbb{Z}/m\mathbb{Z} \times \mathbb{Z}/m\mathbb{Z}$. Thus, in order for R and S to form a basis, we must show $\langle R\rangle \cap \langle S\rangle = \{\mathcal{O}_E\}$.

Suppose $[t]R = [u]S \neq \mathcal{O}_E$ for some integers t, u. By assumption, it must be that $m \nmid t$ and $m \nmid u$. Now consider $e_m([t]R - [u]S, S) = 1$, since $e_m(\mathcal{O}_E, T) = 1$ for any T. By the bilinearity of the pairing, this gives

$$e_m([t]R - [u]S, S) = e_m(R, S)^t e_m(S, S)^{-u}.$$

Then, because $e_m(S, S) = 1$, we arrive at the conclusion $e_m(R, S)^t = 1$, which is a contradiction since $e_m(R, S)$ has order m and $m \nmid t$. Thus, there can exist no such integers t, u, and therefore $\langle R\rangle \cap \langle S\rangle = \{\mathcal{O}_E\}$. □

Definition 1.6 (Independent points, isogenies)**.** When two points $R, S \in E$ satisfy any of the equivalent properties in Lemma 1.5, we say they are **independent** of one another. Similarly, we say that two cyclic groups of order ℓ^e are independent whenever any of their generators are. Finally, we say that two isogenies of degree ℓ^e are independent if their kernels are.

Lemma 1.7. *Let $\phi : E \to E/\langle R\rangle$ be an isogeny of degree ℓ^e with kernel $\langle R\rangle$, and let S be a point of order ℓ^e, such that S and R are independent. Then $\phi(S)$ has order ℓ^e and generates* $\ker(\widehat{\phi})$.

Proof. Because R and S are independent (Definition 1.6), the subgroups generated by R and S intersect trivially. Thus, since ϕ has kernel $\langle R\rangle$, no non-trivial point in $\langle S\rangle$ is in the kernel of ϕ. Furthermore, we know that $\widehat{\phi} \circ \phi = [\ell^e]$ has kernel $E[\ell^e]$, and that $S \in E[\ell^e]$. Thus, $\widehat{\phi}(\phi(S)) = \mathcal{O}$, implying $\phi(S)$ is in the kernel of $\widehat{\phi}$. The same holds for all elements $S' = [\lambda]S \in \langle S\rangle$, and since $\phi(S') \neq \mathcal{O}$ for all non-trivial S', $\phi(S)$ has order ℓ^e and generates $\ker(\widehat{\phi})$. □

The following lemma is the main tool we will use, repeatedly, to design the Proofs of Knowledge in Chapter 4.

Lemma 1.8. *Let ℓ_1, ℓ_2 be distinct primes different from p, and let e_1, e_2 be positive integers. Let $\phi_A : E \to E_A$ be an isogeny of degree $\ell_1^{e_1}$. Furthermore, let $\phi_B : E \to E_B$ and $\phi_{AB} : E_A \to E_{AB}$ be isogenies of degree $\ell_2^{e_2}$ such that* $\ker(\phi_{AB}) = \phi_A(\ker(\phi_B))$. *Then there exists an isogeny $\phi_{BA} : E_B \to E_{AB}$ of degree $\ell_1^{e_1}$.*

Additionally, let $S \in E$ be a point of order $\ell_2^{e_2}$ such that $\ker(\phi_B)$ *and $\langle S\rangle$ are independent, let $S_B = \phi_B(S)$ and let $S_{AB} = \phi_{AB}(\phi_A(S))$. Then S_B and S_{AB} both have order $\ell_2^{e_2}$, and generate, respectively,* $\ker(\widehat{\phi_B})$ *and* $\ker(\widehat{\phi_{AB}})$. *Moreover, $\phi_{BA}(S_B) = S_{AB}$.*

This is visualised in Figure 1.2.

Proof. Let K_A be a generator of $\ker(\phi_A)$. Then, because the degrees of ϕ_A, ϕ_B are coprime, $\phi_B(K_A)$ also has order $\ell_1^{e_1}$ and generates the kernel of some isogeny $\chi : E_B \to E_B/\langle\phi_B(K_A)\rangle$.

Observe that E_{AB} is defined as the codomain of $\phi_{AB} \circ \phi_A$. We thus have that $E_{AB} \cong E/\langle K_A, K' \rangle$ for a point K' of order $\ell_2^{e_2}$ such that $\langle \phi_A(K') \rangle = \ker(\phi_{AB})$. Because $\ker(\phi_{AB}) = \phi_A(\ker(\phi_B))$, we conclude $\langle K' \rangle = \ker(\phi_B)$. Therefore, $E_B/\langle \phi_B(K_A) \rangle \cong E_{AB}$ as required.

By the conditions on S, Lemma 1.7 shows that $S_B = \phi_B(S)$ generates $\ker(\widehat{\phi_B})$.

One can verify that using Vélu's formula [Vél71], $\phi_{AB}(\phi_A(P)) = \phi_{BA}(\phi_B(P))$ for any point $P \in E$ (see [Leo20, Lemma 1]). Hence,

$$\begin{aligned} S_{AB} &= \phi_{AB}(\phi_A(S)) \\ &= \phi_{BA}(\phi_B(S)) \\ &= \phi_{BA}(S_B). \end{aligned}$$

Finally, because $\langle S \rangle$ and $\ker(\phi_B)$ are independent, and because $\ker(\phi_{AB}) = \phi_A(\ker(\phi_B))$, then $\langle \phi_A(S) \rangle$ and $\ker(\phi_{AB})$ must also be independent. $\phi_A(S)$ must have order $\ell_2^{e_2}$ since the degree of ϕ_A is coprime to the order of S. So by applying Lemma 1.7 again, we arrive at the conclusion that S_{AB} generates $\ker(\widehat{\phi_{AB}})$. □

The lemma above suggests an algorithm to compute the points S_B and S_{AB}, even when the isogeny ϕ_A is only known through its action on $E[\ell_2^{e_2}]$. We present such an algorithm in Figure 1.1.

$$\begin{tikzcd}
E, S \in E \arrow[r, "\phi_A"] \arrow[d, "\phi_B"'] & E_A \arrow[d, "\phi_{AB}"] \\
\begin{array}{c} E_B \\ S_B = \phi_B(S) \end{array} \arrow[r, dashed, "\phi_{BA}"] & \begin{array}{c} E_{AB} \\ S_{AB} = \phi_{BA}(S_B) \\ = \phi_{AB}(\phi_A(S)) \end{array}
\end{tikzcd}$$

Figure 1.2: Lemma 1.8, visualised. The lemma shows that ϕ_{BA} exists and the equality on S_{AB} from both directions holds.

Algorithm 1.1 Algorithm to compute image of a single point under hidden isogeny ϕ_{BA}, as per Lemma 1.8.

```
Input: $(E, P, Q, E_A, P_A, Q_A, \phi_B, \phi_{AB})$ such that $\langle P, Q \rangle = E[\ell_2^{e_2}]$, $\phi_B : E \to E_B$ and $\phi_{AB} : E_A \to E_{AB}$ have degree $\ell_2^{e_2}$, and for some isogeny $\phi_A : E \to E_A$ of degree $\ell_1^{e_1}$, we have $\ker(\phi_{AB}) = \phi_A(\ker(\phi_B))$, $P_A = \phi_A(P)$, and $Q_A = \phi_A(Q)$
Output: $(S, \phi_{BA}(S))$ where $S \in E_B$ and $\phi_{BA} : E_B \to E_{AB}$ is an isogeny such that $\ker(\phi_{BA}) = \phi_B(\ker(\phi_A))$
Find a point $S \in E$ such that $S_B = \phi_B(S)$ generates the kernel of $\widehat{\phi_B}$. In fact, it suffices to use either $S = P$ or $S = Q$.
Write $S = [a]P + [b]Q$ for integers $a, b \in \mathbb{Z}/\ell_2^{e_2}\mathbb{Z}$
Compute $S_A = [a]P_A + [b]Q_A$. Then $S_A = \phi_A(S)$ despite $\phi_A$ being unknown.
Compute $S_{AB} = \phi_{AB}(S_A)$
Output $(S_B, S_{AB})$
```

Chapter 2

Cryptographic Hardness Assumptions from Isogenies

Mathematical cryptography promotes rigorous proofs of security for cryptographic primitives and protocols, with reductions to clearly defined assumptions. In this chapter, we tour some well-established hardness assumptions of isogeny-based cryptography, but also visit some new ones.

It is usual to first define a "hard problem", precisely laying out the challenge for adversaries to solve. From a hard problem, a corresponding hardness assumption can be easily given—usually that for certain parameter choices, the aforementioned hard problem is infeasible to solve by any adversary within specified restrictions.

Commonly, problems come in pairs—a computational problem is usually coupled with a decisional variant. Computational problems are the more general class and refer to challenges in which the adversary is required to compute some value (for example, finding a secret key or shared secret). Decisional problems are those where the adversary outputs a binary value stating whether some relationship holds or not (for example, whether or not the shared secret of two given Diffie–Hellman public keys is equal to a specified value).

The foundational hardness assumption of isogeny-based cryptography is that it is hard to find an isogeny between two given curves. This is formalised as follows:

Definition 2.1 (General isogeny problem)**.** Given j-invariants $j, j' \in \mathbb{F}_q$, find an isogeny $\phi : E \to E'$ if one exists, where $j(E) = j$ and $j(E') = j'$.

We could also define a decisional variant of this problem: given two curves, to determine whether there exists an isogeny between them. However, this decisional version is easy to solve—an isogeny exists over $\mathbb{F}_q$ if and only if $\#E(\mathbb{F}_q) = \#E'(\mathbb{F}_q)$, as we saw in Chapter 1, and the cardinality of an elliptic curve can be computed very efficiently [Sch95] (we will discuss this more in Chapter 7). By [Sil09, Exercise 5.13], the cardinality of E and E' over field extensions $\mathbb{F}_{q^k}$ are determined by their cardinalities over $\mathbb{F}_q$.

In the following sections, we will turn to more specialised isogeny-based assumptions that will appear throughout this thesis and in a variety of schemes in the literature.

2.1 Standard SIDH hardness assumptions

Of particular importance in this thesis are the cryptographic hardness assumptions specific to the SIDH protocol, discussed in Section 1.3. Throughout this section, let pp denote the SIDH public parameters $\mathsf{pp} = (\ell_1, \ell_2, e_1, e_2, p, E_0, P_1, Q_1, P_2, Q_2)$. We begin with the indispensable assumption that a secret key (isogeny) cannot be recovered from an SIDH public key.

Definition 2.2 (Computational Supersingular Isogeny (CSSI) problem)**.** For fixed SIDH public parameters pp, let $\phi : E_0 \to E'$ be an isogeny of degree $\ell_1^{e_1}$, and let $P' = \phi(P_2), Q' = \phi(Q_2)$. Given the SIDH public key (E', P', Q'), find an isogeny $\phi' : E_0 \to E'$ of degree $\ell_1^{e_1}$ such that $P' = \phi'(P_2)$ and $Q' = \phi'(Q_2)$.

This is problem 5.2 of [DJP14] and is also called the SIDH isogeny problem by Galbraith and Vercauteren [GV18, Definition 2]. Note that we lose no generality by writing of ϕ having degree $\ell_1^{e_1}$, because the SIDH public parameters are symmetric and $\ell_1^{e_1}$ and $\ell_2^{e_2}$ are interchangeable.

In many cryptosystems, it is possible to verify that a given public key is well-formed or valid, without knowledge of the corresponding secret. For example, in elliptic curve cryptography, it is simple to validate that a (public key) point P does indeed lie on a given curve E, and that it lies in the required subgroup of the curve (in cases where we use a curve with a non-trivial cofactor). In supersingular isogeny-based cryptography, it is possible to confirm that a given curve is supersingular. The SIDH protocol, however, is in a rather unique predicament. Not only is there no known way to verify that a given SIDH public key is well-formed, it is in fact an established cryptographic assumption that such key validation is infeasible, as we shall now see.

The specific issue with SIDH key validation is that, given a public key (E', P', Q'), we cannot validate that P', Q' are actually the correct images of basis points P_2, Q_2 under the secret isogeny ϕ. This difficulty lies at the heart of the adaptive attacks discussed in Chapter 3. The best we can do is to check that the points are indeed a basis of the correct order $\ell_2^{e_2}$, and use the Weil pairing check which will be discussed in Section 3.3:

$$e_{\ell_2^{e_2}}(P', Q') = e_{\ell_2^{e_2}}(P_2, Q_2)^{\deg(\phi)}. \tag{2.1}$$

The following decisional problem captures this obstruction to validation and takes after Definition 3 of [GV18]. It is also very similar to the key validation problem of Urbanik and Jao [UJ18, Problem 3.4] (the key validation problem asks whether a ϕ of degree *dividing* $\ell_1^{e_1}$ exists). However, the previous definitions did not take the Weil pairing check into account, which would serve as a distinguisher. We include this explicitly in our definition.

Definition 2.3 (Decisional SIDH isogeny (DSIDH) problem)**.** For fixed SIDH public parameters pp, the DSIDH problem is to distinguish between the following two distributions:

- $\mathcal{D}_0 = \{(E', P', Q')\}$ such that $\phi : E_0 \to E'$ is an isogeny of degree $\ell_1^{e_1}$, $P' = \phi(P_2)$, and $Q' = \phi(Q_2)$.
- $\mathcal{D}_1 = \{(E', P', Q')\}$ such that E' is any supersingular elliptic curve over $\mathbb{F}_{p^2}$ with the same cardinality as E_0, and P', Q' is a basis of $E'[\ell_2^{e_2}]$ satisfying the Weil pairing check $e_{\ell_2^{e_2}}(P', Q') = e_{\ell_2^{e_2}}(P_2, Q_2)^{\ell_1^{e_1}}$.

Remark 2.4. As shown by Galbraith and Vercauteren [GV18], Thormarker [Tho17], and Urbanik and Jao [UJ18], being able to solve this decisional problem is as hard as solving the computational (CSSI) problem, so key validation is fundamentally difficult. This is done by testing ℓ_1-isogeny neighbouring curves of E_1 and learning the correct path one bit at a time (note that this requires a perfect decision oracle). We refer the reader to [GV18] for details.

Typically, a computational problem that allows the adversary access to an oracle solving the corresponding decisional problem is called a **gap** problem (since it relies on there being a *gap* between solving the decisional and the computational versions). The reduction from CSSI to DSIDH shows the gap problem here is easy.

The previous two assumptions are concerned with a single SIDH public key and corresponding private isogeny. Now, we turn to assumptions about the full SIDH protocol and its commutative diagram. The following is Problem 5.5 of [DJP14], and intuitively states that it is hard to determine whether there exist valid "vertical sides" to an SIDH square, given the corners and the bottom horizontal side.

Definition 2.5 (Decisional Supersingular Product (DSSP) problem)**.** Let $\phi : E_0 \to E_1$ be an isogeny of degree $\ell_1^{e_1}$. Let $P_2, Q_2 \in E_0[\ell_2^{e_2}]$ be a fixed basis of the $\ell_2^{e_2}$-torsion subgroup. Suppose we have the following two distributions:

- $\mathcal{D}_0 = \{(E_2, E_3, \phi')\}$ such that there exists a cyclic subgroup $G \subseteq E_0[\ell_2^{e_2}]$ of order $\ell_2^{e_2}$ with $E_2 \cong E_0/G$ and $E_3 \cong E_1/\phi(G)$, and where $\phi' : E_2 \to E_3$ is a degree-$\ell_1^{e_1}$ isogeny.
- $\mathcal{D}_1 = \{(E_2, E_3, \phi')\}$ such that E_2 is a random supersingular curve with the same cardinality as E_0, and E_3 is the codomain of a random isogeny $\phi' : E_2 \to E_3$ of degree $\ell_1^{e_1}$.

Let $\mathcal{O}^{\mathsf{DSSP}}$ be an oracle that behaves as follows: on setup, with public parameters $(E_0, P_2, Q_2, E_1, \phi(P_2), \phi(Q_2))$, it chooses a uniformly random secret bit $b \leftarrow \{0, 1\}$. Each time it is queried, $\mathcal{O}^{\mathsf{DSSP}}$ returns a tuple from distribution $\mathcal{D}_b$. The DSSP problem is then, given access to such an oracle, to determine b.

Finally, we come to a pair of problems related to the shared secret of an SIDH key exchange, rather than the public keys involved.

Definition 2.6 (Computational Supersingular Isogeny Diffie–Hellman (SI-CDH) problem)**.** Let pp be SIDH public parameters, and

$$\begin{aligned} K_1 &= (E_1, \phi_1(P_2), \phi_1(Q_2))\,, \\ K_2 &= (E_2, \phi_2(P_1), \phi_2(Q_1))\,, \end{aligned}$$

be two SIDH public keys, where $\phi_i : E_0 \to E_i$ has degree $\ell_i^{e_i}$. The SI-CDH problem is, given pp, K_1, and K_2, to compute the j-invariant $j = \mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2)$.

We define the advantage of a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ solving the SI-CDH problem as

$$\mathsf{Adv}^{\text{si-cdh}}(\mathcal{A}) = \Pr\left[\, j' = \mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2) \;\middle|\; j' \leftarrow \mathcal{A}(\mathsf{pp}, K_1, K_2) \,\right].$$

The SI-CDH assumption states that for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{\text{si-cdh}}(\mathcal{A}) \leq \mathsf{negl}$. In other words, given the two keys involved in an SIDH exchange, it should be infeasible to compute the resulting shared secret of the exchange. This is analogous to the computational Diffie–Hellman (CDH) problem in classical cryptography, of computing the shared secret $g^{ab} \pmod p$ given the two public keys g^a and g^b. The corresponding decisional problem follows.

For ease of notation, we define

$$\mathrm{SSJ}_{\mathsf{pp}} = \big\{ j(E_i) \,:\, E_i \text{ defined over } \mathbb{F}_q \text{ and supersingular} \big\} \tag{2.2}$$

to be the set of all supersingular j-invariants over the field $\mathbb{F}_q$ established by the public parameters pp. Every shared secret arising from an SIDH key exchange with public parameters pp is therefore contained in this set.

Definition 2.7 (Decisional Supersingular Isogeny Diffie–Hellman (SI-DDH) problem)**.** Let pp be SIDH public parameters. Define two distributions:

- $\mathcal{D}_0 = \{(K_1, K_2, j)\}$ such that $j = \mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2)$,
- $\mathcal{D}_1 = \{(K_1, K_2, j)\}$ such that $j \leftarrow \mathrm{SSJ}_{\mathsf{pp}}$,

where in both cases

$$\begin{aligned} K_1 &= (E_1, \phi_1(P_2), \phi_1(Q_2))\,, \\ K_2 &= (E_2, \phi_2(P_1), \phi_2(Q_1))\,, \end{aligned}$$

are two SIDH public keys, and ϕ_i has degree $\ell_i^{e_i}$. The SI-DDH problem is to distinguish between the distributions $\mathcal{D}_0$ and $\mathcal{D}_1$.

We define the advantage of a PPT adversary $\mathcal{A}$ solving the SI-DDH problem as

$$\mathsf{Adv}^{\text{si-ddh}}(\mathcal{A}) = \left| \Pr\left[b = b' \mid b' \leftarrow \mathcal{A}(\mathsf{pp}, K_1, K_2, j_b),\ b \leftarrow \{0,1\} \right] - \frac{1}{2} \right|,$$

where $(K_1, K_2, j_b) \in \mathcal{D}_b$. The SI-DDH assumption states that for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{\text{si-ddh}}(\mathcal{A}) \leq \mathsf{negl}$.

Remark 2.8 (SI-GDH Gap assumption)**.** From Remark 2.4, we know that an oracle solving the decisional SIDH (DSIDH) problem can be used to solve the computational CSSI problem. Fujioka, Takashima, Terada, and Yoneyama [FTTY18] show a similar result holds for what they call the *degree-sensitive* supersingular isogeny gap Diffie-Hellman (ds-SI-GDH). They show that, utilising an oracle which perfectly distinguishes the existence of correct-degree isogenies between the four "corners" of an SIDH commutative square, the SI-CDH problem can be solved efficiently by an adversary—thus making the ds-SI-GDH problem easy. We refer the reader to [FTTY18, Proposition 1] for details.

2.2 A degree-insensitive assumption

In this section, we will analyse a novel cryptographic assumption introduced by Fujioka et al. [FTTY18]. They name this assumption the **degree-insensitive supersingular isogeny gap Diffie–Hellman** (di-SI-GDH) assumption, and use it in a proof of security for their biclique SIDH protocol—an authenticated key exchange protocol from isogenies.

Fujioka et al. introduce this degree-insensitive gap assumption in an attempt to avoid the lack of traditional gap assumptions for SIDH, relating to Remark 2.8. Specifically, reductions such as the one by Galbraith and Vercauteren [GV18] from the CSSI to the DSIDH problem rely on detecting whether the specific degrees of the isogenies are correct. Therefore, removing any restriction on these degrees is a potential avenue for breaking the reduction and forming a "gap".

We conjecture that this assumption is invalid. When one considers the SIDH commutative diagram in Figure 1.1, the isomorphism class of the curve E_{AB} is uniquely determined by the choice of secret kernels (or, equivalently, isogenies) by the participants A and B. In fact, an important characteristic of the isogenies used in SIDH is that the degrees of these isogenies ϕ_A, ϕ_B (both approximately $\sqrt{p}$) are much smaller than the number of isomorphism classes of supersingular elliptic curves (which, recall, is roughly $p/12$). This means that the public curves (E_0, E_A, E_B) uniquely determine the shared curve E_{AB} (of course, this is still hard to compute without secret information). Thus, the j-invariant of the final curve produced by both parties is able to function as a unique shared secret. If the use of larger-degree isogenies is allowed, then eventually (the isomorphism class of) E_{AB} is no longer uniquely determined by E_A and E_B. The natural question then, directly related to the degree-insensitive oracle of [FTTY18], is whether the provision of torsion generator images provides enough additional restriction to uniquely define E_{AB}. We investigate this question now, starting with some useful definitions and formal conjectures.

Definitions and conjectures

We define some notation for clarity and brevity. For a participant $i \in \{1, 2\}$ in an SIDH key exchange, we let $\underline{i} = (3 - i) \in \{2, 1\}$ represent the other participant.

Let

$$\mathrm{SSEC}_{\mathsf{pp},i} = \left\{(E_i,\, \phi_i(P_{\underline{i}}),\, \phi_i(Q_{\underline{i}})) \;\middle|\; \phi_i : E_0 \to E_i,\ \deg(\phi_i) = \ell_i^{e_i}\right\} \tag{2.3}$$

be the set of all **valid** public keys for participant i (formed from isogenies of degree $\ell_i^{e_i}$) in the SIDH protocol with public parameters pp.

We now define two additional sets which will be used in the forthcoming discussion. The first,

$$\mathrm{SSEC}^*_{\mathsf{pp},i} = \left\{(E', P', Q') \;\middle|\; E' \text{ supersingular},\ P', Q' \in E',\ \langle P', Q'\rangle = E'[\ell_{\underline{i}}^{e_{\underline{i}}}]\right\}, \tag{2.4}$$

is the set of all possible triples that take the same form as SIDH public keys. Note that this definition is completely independent of any isogeny from E_0 to E'. We then have that $\mathrm{SSEC}_{\mathsf{pp},i} \subseteq \mathrm{SSEC}^*_{\mathsf{pp},i}$.

As was mentioned above, and will be discussed in more detail in Section 3.3, the Weil pairing implies an extra condition on the points in SIDH public keys. Specifically, for an isogeny $\phi : E_0 \to E_{\underline{i}}$, we can check that

$$e_{\ell_i^{e_i}}(\phi(P_i), \phi(Q_i)) = e_{\ell_i^{e_i}}(P_i, Q_i)^{\deg(\phi)} \tag{2.5}$$

This is a simple additional condition that can be applied to distinguish between $\mathrm{SSEC}_{\mathsf{pp},i}$ and $\mathrm{SSEC}^*_{\mathsf{pp},i}$ (and similarly between χ and χ^{ds} or χ^{di}, defined below). In many cases, such as those we will see in Section 2.2.2, this in fact provides no extra restriction when $\deg(\phi)$ can be arbitrary powers of ℓ_1. In cases where it does matter, the Weil pairing check can be simply incorporated into the definitions of SSEC (and χ), reducing the sizes of these sets. We thus ignore it for the sake of simplicity in the following exposition, observing that a distinguisher between these modified sets functions in exactly the same manner and that the pairing is easily computed.

Finally, we define the set of **degree-insensitive** SIDH public keys, following the definition of the di-SI-GDH oracle by Fujioka et al. [FTTY18].

$$\mathrm{SSEC}^{di}_{\mathsf{pp},i} = \left\{ (E_i,\, \phi_i(P_{\underline{i}}),\, \phi_i(Q_{\underline{i}}) \;\middle|\; \phi_i : E_0 \to E_i,\ \deg(\phi_i) = \ell_i^m,\ m \in \mathbb{N} \right\}. \tag{2.6}$$

We have, then, that $\mathrm{SSEC}_{\mathsf{pp},i} \subseteq \mathrm{SSEC}^{di}_{\mathsf{pp},i} \subseteq \mathrm{SSEC}^{*}_{\mathsf{pp},i}$. As we shall soon discuss, the degree restraint in the definition of $\mathrm{SSEC}_{\mathsf{pp},i}$ uniquely determines the ϕ_i for each tuple. In the case of $\mathrm{SSEC}^{di}_{\mathsf{pp},i}$, however, it is one of our main conjectures that the ϕ_i are not uniquely determined by the triples—that there are multiple isogenies of varying degrees that all produce the same public key tuple.

Now that we have defined these three sets, we shall define corresponding "SIDH square" sets. The first of these is the weakest case,

$$\chi_{\mathsf{pp}} = \left\{ (K_1, K_2, j) \;\middle|\; K_i \in \mathrm{SSEC}^{*}_{\mathsf{pp},i},\ j \in \mathrm{SSJ}_{\mathsf{pp}} \right\}. \tag{2.7}$$

These triples form *possible* corners of SIDH squares (that is, public keys and shared secret), but there do not necessarily exist any particular isogenies between these corners. We then have the degree-sensitive and degree-insensitive restricted versions of this set:

$$\chi^{ds}_{\mathsf{pp}} = \left\{ \left(K_1, K_2, \mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2)\right) \;\middle|\; K_1 \in \mathrm{SSEC}_{\mathsf{pp},1},\ K_2 \in \mathrm{SSEC}_{\mathsf{pp},2} \right\}, \tag{2.8}$$

and

$$\chi^{di}_{\mathsf{pp}} = \left\{ \left(K_1, K_2, \mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2)\right) \;\middle|\; K_1 \in \mathrm{SSEC}^{di}_{\mathsf{pp},1},\ K_2 \in \mathrm{SSEC}^{di}_{\mathsf{pp},2} \right\}, \tag{2.9}$$

with elements of these sets each corresponding to a valid SIDH square such as Figure 2.1, in the degree-sensitive and degree-insensitive setting respectively. The second set simply loosens the restriction on the isogeny degrees, so that $\deg(\phi_1) = \deg(\phi_1') = \ell_1^m$ for any $m \in \mathbb{N}$, and likewise $\deg(\phi_2) = \deg(\phi_2') = \ell_2^n$ for any $n \in \mathbb{N}$. We necessarily have that $\chi^{ds}_{\mathsf{pp}} \subseteq \chi^{di}_{\mathsf{pp}} \subseteq \chi_{\mathsf{pp}}$.

$$\begin{tikzcd}
E_0 \arrow[r, "\phi_1"] \arrow[d, "\phi_2"'] & K_1 \arrow[d, "\phi_2'"] \\
K_2 \arrow[r, "\phi_1'"'] & E'
\end{tikzcd}$$

Figure 2.1: A re-labelling of the SIDH commutative diagram from Figure 1.1, where $j(E') = \mathrm{SIDH}(K_1, K_2)$, $\ker(\phi_1') = \phi_2(\ker(\phi_1))$ and $\ker(\phi_2') = \phi_1(\ker(\phi_2))$.

The degree-insensitive SI-GDH oracle defined by Fujioka et al. [FTTY18] distinguishes between the sets χ_{pp} and χ^{di}_{pp}, while the degree-sensitive variant of the oracle distinguishes between χ_{pp} and χ^{ds}_{pp}. We conjecture that in the former (degree-insensitive) case, $\chi_{\mathsf{pp}} = \chi^{di}_{\mathsf{pp}}$. That is, all possible pairs of torsion basis points on every possible supersingular elliptic curve, and all final shared j-invariants, can be produced by suitable degree-insensitive choices of isogenies ϕ_1 and ϕ_2. The aim of the rest of this section is to provide some evidence for this conjecture.

Fujioka et al. hint toward the possibility of this problem, stating:

> *Therefore, as an extreme possible case, any tuple of supersingular elliptic curves (E_A, E_B, E_{AB}) might form the commutative diagram in [Figure 1.1], that is, any tuple of such curves would be true instances in the hypothetical case. We cannot exclude such possibility from our present knowledge of the di-SI-GDH problem.*

We conjecture a much stronger result, however, by including all pairs of points in our claim too. We proceed in two stages. In the first, we will give evidence that all possible SIDH public key tuples in $\mathrm{SSEC}^{*}_{\mathsf{pp},i}$ do actually arise as valid public key tuples (when considering all possible choices of degree-insensitive ϕ_i). This is Conjecture 2.9.

Conjecture 2.9. $\mathrm{SSEC}^{*}_{\mathsf{pp},i} = \mathrm{SSEC}^{di}_{\mathsf{pp},i}$

We then give evidence that for any two such public key tuples $K_1 \in \mathrm{SSEC}^{*}_{\mathsf{pp},1}, K_2 \in \mathrm{SSEC}^{*}_{\mathsf{pp},2}$, and for any choice of supersingular curve E', then $\big(K_1, K_2, j(E')\big) \in \chi^{di}_{\mathsf{pp}}$. In other words, any supersingular j-invariant is a valid shared secret and may be produced in a degree-insensitive key exchange with any pair of keys. This is because the loosened restriction on the degrees of the isogenies ϕ_i is not enough to uniquely determine the isomorphism class of the fourth curves E_{AB}, owing to the fact that there exist many different isogenies with different kernels which produce identical public key tuples. This is summarised in Conjecture 2.10.

Conjecture 2.10. $\chi_{\mathsf{pp}} = \chi^{di}_{\mathsf{pp}}$

2.2.1 Uniqueness of isogenies from public keys

Recall that the SIDH protocol parameters are chosen so that $\ell_1^{e_1} \approx \ell_2^{e_2}$ (referred to as **balanced** parameters). When this is the case, the public keys used by each participant uniquely determine the secret isogenies, as shown by Martindale and Panny [MP19].

Theorem 2.11. *Two distinct isogenies $\phi, \varphi : E \to E'$ of degree d have distinct actions on the m-torsion $E[m]$ if $m^2 > 4d$.*

Proof (due to [MP19]). Suppose ϕ, φ have the same action on $E[m]$. Then $\ker(\phi - \varphi) \supseteq E[m]$. This implies that $\deg(\phi - \varphi) \geq \# \ker(\phi - \varphi) \geq m^2$. We also have, by [Sil09, Lemma V.1.2], that $\deg(\phi - \varphi) \leq 4d$. Thus, we must have that $m^2 \leq 4d$. If $m^2 > 4d$, the actions of ϕ and φ must therefore be distinct. □

Precisely, unless $\ell_2^{2e_2} \leq 4\ell_1^{e_1}$, a degree-$\ell_1^{e_1}$ isogeny is uniquely defined by its action on the $\ell_2^{e_2}$-torsion. The analogue for $\ell_2^{e_2}$-isogenies also holds, by the same token. It follows that the shared secret is also uniquely defined (although, of course, still hard to compute without knowledge of either of the secret keys). We now show that, with balanced SIDH parameters, even if (only) one participant uses a degree-insensitive isogeny, the two public keys still uniquely determine the shared secret both parties derive.

Lemma 2.12. *Let* pp *be standard (balanced) SIDH protocol parameters. If participant i uses an isogeny ϕ_i of (degree-sensitive) degree $\ell_i^{e_i}$, then the SIDH shared secret $j(E_{AB})$ is uniquely determined regardless of the other participant's isogeny degree, $\deg(\phi_{\underline{i}}) = \ell_{\underline{i}}^{n}$ for $n \in \mathbb{N}$.*

Proof. Without loss of generality, assume that it is participant B who uses an isogeny ϕ_B of correct degree $\ell_2^{e_2}$. We refer the reader to Figure 2.2 to set some notation. As usual, we have an elliptic curve E_0 along with a chosen $\ell_2^{e_2}$-torsion basis P_2, Q_2. Consequently, ϕ_B has kernel $\langle K_B = P_2 + [\beta]Q_2 \rangle$ for some secret β. Suppose that we have two isogenies $\phi_A, \phi'_A : E_0 \to E_A$ such that

$$\begin{aligned} P' &= \phi_A(P_2) = \phi'_A(P_2), \\ Q' &= \phi_A(Q_2) = \phi'_A(Q_2). \end{aligned}$$

Denote generators of the kernels of these maps by K_A, K'_A respectively (so clearly $E/\langle K_A \rangle \cong E/\langle K'_A \rangle$). Now, irrespective of which isogeny participant A used, B will compute the isogeny ϕ_{AB} with kernel

$$\ker(\phi_{AB}) = \langle P' + [\beta]Q' \rangle$$

because B's computation is only based on A's public key (E_A, P', Q'). Therefore, E_{AB} is uniquely determined (up to isomorphism) as

$$E_0/\langle K_A, K_B \rangle \cong E_0/\langle K'_A, K_B \rangle \cong E_{AB}.$$

□

$$\begin{tikzcd}
E_0 \arrow[r, bend left, "\phi_A"] \arrow[r, bend right, "\phi'_A"'] \arrow[d, "\phi_B"'] & E_A \arrow[d, "\phi_{AB}"] \\
E_B \arrow[r, bend left, "\phi_{BA}"] \arrow[r, bend right, "\phi'_{BA}"'] & E_{AB}
\end{tikzcd}$$

Figure 2.2: Commutative diagram of Lemma 2.12, where $\ker(\phi_{BA}) = \phi_B(\ker(\phi_A))$, $\ker(\phi'_{BA}) = \phi_B(\ker(\phi'_A))$ and $\ker(\phi_{AB}) = \phi_A(\ker(\phi_B)) = \phi'_A(\ker(\phi_B))$.

We thus require for Conjecture 2.10, that both isogenies be of arbitrary-power degree. It is clearly not sufficient for only one of the two isogeny choices to be degree-insensitive.

2.2.2 Experimental evidence

In this section, we exhibit the experimental evidence we have collected for Conjectures 2.9 and 2.10.

Evidence that all public key tuples are valid

In the case that $p = 2^3 3^2 - 1 = 71$, we have shown using `MAGMA` that using arbitrary paths of 2-isogenies, we can find paths that give, as torsion point images, any pair of points P, Q which generate $E[3^2]$ on any elliptic curve E in the isogeny graph (up to isomorphism). There are exactly 3888 unique ordered tuples of points P, Q on each elliptic curve $E(\mathbb{F}_{p^2})$ such that both P and Q have order 3^2, and $\langle P, Q \rangle = E[3^2]$ (by Equation 3.15). In order to accommodate for isomorphism classes of the curves, one representative curve was (arbitrarily) chosen for each j-invariant. Then, for all subsequent curves with the same j-invariant found by the algorithm, the isomorphism to the representative curve was computed and used to translate the points to this representative curve. This allowed for the removal of duplicate public key tuples. We found that all 3888 unique ordered pairs on all seven isogenous supersingular curves were reached from the starting P_2, Q_2 on the curve E_0 with j-invariant 0.

We have also verified this result with high probability over a larger field, where $p = 2^2 3^3 - 1 = 107$. In this case, there are exactly 314,928 possible ordered pairs P, Q generating $E[3^3]$ on each supersingular elliptic curve E (again, by Equation 3.15). We allowed our simulation to run until the points reached on the curve with j-invariant 94 totalled 314,673, which is approximately 99.92% of the expected total number of points. Due to the probabilistic nature of the algorithm, the discovery of new pairs of points decreases rapidly as the number of duplicates increases, so we decided to end the simulation early (after a total of 20 million 2-isogenies had been traversed in the supersingular isogeny graph), deciding that this was overwhelming evidence that all points were likely reachable as in the smaller case. We see no reason that these results would not extend to supersingular curves over finite fields of any choice of p, as required by Conjecture 2.9.

In both of these cases, the Weil pairing imposes no extra restriction on the validity of a randomly chosen pair of independent points for each choice of isogeny. Observe that for $N = 3^2$, 2 has order 6 modulo 9, and for $N = 3^3$, 2 has order $18 = \varphi(3^3)$, so there is no apparent theoretical reason why not all choices would be valid—as we observe in practice.

Evidence that all j-invariants are valid given any public keys

We now discuss Conjecture 2.10. This second conjecture relies on Conjecture 2.9, because if it were possible to distinguish $\mathrm{SSEC}^{*}_{\mathsf{pp},i}$ from $\mathrm{SSEC}^{di}_{\mathsf{pp},i}$, the same distinguisher could be applied to the public keys in χ_{pp} to distinguish it from χ^{di}_{pp}. Furthermore, Conjecture 2.10 additionally requires that the j-invariant $j(E_{AB})$ should not provide any advantage in distinguishing between the two sets.

We make an important observation, that when the degree of the isogeny is allowed to be an arbitrary power of ℓ_i, the image of points $P_{\underline{i}}, Q_{\underline{i}}$ no longer uniquely determines the isogeny used. That is, not only does there exist a power-of-ℓ_i isogeny ϕ_i for any pair P', Q' such that $P' = \phi_i(P_{\underline{i}}), Q' = \phi_i(Q_{\underline{i}})$, there exist *many* such isogenies—each with a different kernel.

In order to demonstrate this in practice using `MAGMA`, we used extension fields to define points of higher order as the kernels of degree-insensitive isogenies. For example, we used $\mathbb{F}_{p^6}$ when $p = 71$ as above, to find a point of order 27 generating the kernel of an isogeny ϕ_B.

Let $\alpha \in \mathbb{F}_{p^6}$ denote the element which generates the extension $\mathbb{F}_{p^6}$ over the base field $\mathbb{F}_p$. We begin with elliptic curve $E_0 : y^2 + y = x^3$ with j-invariant 0. In the simulation we selected at random the points

$$\begin{aligned} P_B = \;& (7\alpha^5 + 24\alpha^4 + 49\alpha^3 + 68\alpha^2 + 2\alpha + 5 : \\ & \quad 46\alpha^5 + 36\alpha^4 + 38\alpha^3 + 31\alpha^2 + 3\alpha + 38 : 1), \\ Q_B = \;& (41\alpha^5 + 29\alpha^4 + 3\alpha^3 + 23\alpha^2 + 32\alpha + 56 : \\ & \quad 9\alpha^5 + 41\alpha^4 + 63\alpha^3 + 57\alpha^2 + 33\alpha + 1 : 1), \end{aligned} \tag{2.10}$$

as a basis for the $\ell_2^{e_2}$-torsion on E_0.

We also selected a point of order 27 to define the kernel of the isogeny chosen by participant B (which must be of degree higher than 3^2 by Lemma 2.12):

$$\begin{aligned} K_B = \ & (3\alpha^5 + 41\alpha^4 + 18\alpha^3 + 4\alpha^2 + 13\alpha + 45 : \\ & 27\alpha^5 + 57\alpha^4 + 49\alpha^3 + 11\alpha^2 + 65\alpha + 64 : 1). \end{aligned} \tag{2.11}$$

We proceeded via breadth-first search along non-backtracking 2-isogenies in the isogeny graph to find two distinct isogenies $\phi_A, \phi'_A : E_0 \to E_A$, such that $\phi_A(P_B) = \phi'_A(P_B)$ and $\phi_A(Q_B) = \phi'_A(Q_B)$. Note that non-backtracking means that no 2-isogeny step returned along its dual in the next step, ensuring that distinct paths will compose to produce isogenies with different kernels (and therefore, isogenies which are not equivalent). We present here an example where E_A is the curve $y^2 + y = x^3 + 46x + 60$ with j-invariant 66, and $\deg(\phi_A) = \deg(\phi'_A) = 2^{10}$:

$$\begin{aligned} \phi_A(P_B) = \phi'_A(P_B) = \ & (28\alpha^5 + 25\alpha^4 + 54\alpha^3 + 59\alpha^2 + 8\alpha + 66 : \\ & 66\alpha^5 + 64\alpha^4 + 36\alpha^3 + 63\alpha^2 + 29\alpha + 23 : 1), \\ \phi_A(Q_B) = \phi'_A(Q_B) = \ & (21\alpha^5 + \alpha^4 + 5\alpha^3 + 62\alpha^2 + 6\alpha + 66 : \\ & 58\alpha^5 + 67\alpha^4 + 51\alpha^3 + 36\alpha^2 + 47\alpha + 48 : 1). \end{aligned} \tag{2.12}$$

Simply showing that there exist isogenies with different kernels which produce the same public key is an interesting observation. But to properly support Conjecture 2.10, we wish to demonstrate that these different kernels can produce many (in fact, all) different curves E_{AB}.

The image of the ϕ_B kernel generator K_B under each of these isogenies ϕ_A, ϕ'_A is:

$$\begin{aligned} \phi_A(K_B) = \ & (44\alpha^5 + 39\alpha^4 + 59\alpha^3 + 41\alpha^2 + 55\alpha + 64 : \\ & 7\alpha^5 + 39\alpha^4 + 61\alpha^3 + 64\alpha^2 + 14\alpha + 47 : 1), \\ \phi'_A(K_B) = \ & (67\alpha^5 + 49\alpha^4 + 11\alpha^3 + 19\alpha^2 + 53\alpha + 49 : \\ & 40\alpha^5 + 65\alpha^4 + 13\alpha^3 + 24\alpha^2 + 12\alpha + 67 : 1). \end{aligned} \tag{2.13}$$

Finally, these two images are used to create degree-27 isogenies ϕ_{AB}, ϕ'_{AB} to complete the SIDH square. The first determines an isogeny to the isomorphism class of curves with j-invariant 17, while the second arrives at j-invariant 48. Thus, despite both ϕ_A, ϕ'_A giving the same public key triple, they each determine a different curve E_{AB} in this degree-insensitive key exchange. This shows that E_{AB} is not uniquely determined and strongly supports Conjecture 2.10. To demonstrate that there exist such "collisions" which produce *any* supersingular j-invariant would be computationally intensive in practice, but the presence of at least one such case is strong evidence for our conjecture.

2.2.3 The di-SI-GDH oracle

Above, we have discussed the particular problem of distinguishing between the set of all valid SIDH commutative diagrams χ^{ds}_{pp} or χ^{di}_{pp}, and the set χ_{pp} of all tuples of supersingular elliptic curves with all possible choices of points on them (subject to the easy-to-compute restrictions mentioned earlier), and all possible shared secret j-invariants. In this section, we briefly relate this to the specific working of the di-SI-GDH oracle of Fujioka et al. [FTTY18]. The oracle receives public SIDH parameters pp including the curve E_0, points $P_1, Q_1, P_2, Q_2 \in E_0$, and (candidate) public key tuples of party A and B, along with a j-invariant, j', representing the shared secret obtained through the SIDH protocol with these keys. The oracle then returns 1 if isogenies exist between E_0, E_A, E_B, and a curve E_{AB} such that the public points are mapped in the correct way and the j-invariant of E_{AB} is equal to j' (that is, the provided information forms a valid SIDH square as in Figure 1.1). Otherwise, the oracle returns 0. This oracle is used in Fujioka et al.'s security proof for their authenticated key exchange scheme, to allow a correct simulation in the random oracle model.

Consider, firstly, the public key triples (E', P', Q') provided. Assuming the points obey the subgroup generation and Weil pairing restrictions, Conjecture 2.9 claims that these provide no way to distinguish between $\mathrm{SSEC}^*_{\mathsf{pp},i}$ and $\mathrm{SSEC}^{di}_{\mathsf{pp},i}$, which are equal sets. In addition to this, we give evidence to support the claim (Conjecture 2.10) that even by fixing a choice of two public keys, any choice of j-invariant j' arises from some degree-insensitive choice of isogenies to the fixed keys. We have given experimental evidence that there exist many different isogenies that produce any public key, each with a different kernel. We conjecture that because the kernels of these isogenies uniquely determine the final elliptic curve E_{AB} in the SIDH protocol, but because these kernels are not determined by the public keys, any j-invariant would be a valid shared secret for any choice of two public keys.

Thus, we conjecture that the degree-insensitive SI-GDH oracle as a distinguisher between χ_{pp} and χ^{di}_{pp} cannot exist, because these sets are equal. Hence, we believe that the security proof of Fujioka et al. [FTTY18] is not correct. We stress that this does not mean the protocol in [FTTY18] is broken, only that its security is not justified by the computational assumptions in the paper.

2.3 New SIDH hardness assumptions

We now define the new hardness assumptions which we employ later in this thesis. These hardness assumptions come into play when we reach Chapters 4 and 5, and are used in proving security and zero-knowledge of the protocols we propose. The first of these can intuitively be seen as a "parallel" version of the DSIDH assumption above. In the DSIDH problem, we are trying to determine whether an isogeny exists on the "top" horizontal side of an SIDH square, while our forthcoming new problem does the same for the "bottom" side. We thus call it the decisional *mirror* SIDH problem.

Definition 2.13 (Decisional Mirror SIDH (DMSIDH) problem)**.** Let E_0 and basis points P_2, Q_2 for the $\ell_2^{e_2}$-torsion subgroup $E_0[\ell_2^{e_2}]$ be fixed public parameters, and let $\phi : E_0 \to E_1$ be a randomly sampled isogeny of degree $\ell_1^{e_1}$. Define distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ as follows: construct a random SIDH square by letting $\psi : E_0 \to E_2$ be a random isogeny of degree $\ell_2^{e_2}$, and $\psi' : E_1 \to E_3$ an isogeny of degree $\ell_2^{e_2}$ whose kernel is $\phi(\ker(\psi))$. Next, let $\phi' : E_2 \to E_3$ be an isogeny of degree $\ell_1^{e_1}$ whose kernel is $\psi(\ker(\phi))$. Finally, construct a basis S, T of $E_2[\ell_2^{e_2}]$ with $\langle S \rangle = \ker(\widehat{\psi})$, and let the distributions be:

- $\mathcal{D}_0 = \{(\psi, \psi', S, T, \phi'(S), T')\}$ where $T' = \phi'(T)$
- $\mathcal{D}_1 = \{(\psi, \psi', S, T, \phi'(S), T')\}$ where $T' = \phi'(T + [r]S)$, and r is random.

Let $\mathsf{Gen}^{\mathsf{DMSIDH}}$ be a randomised DMSIDH instance generation algorithm which, given public parameters (E_0, P_2, Q_2), samples a random ϕ and a uniformly random secret bit $b \leftarrow \{0, 1\}$, and returns $(E_1, \phi(P_2), \phi(Q_2))$ and an oracle $\mathcal{O}^{\mathsf{DMSIDH}}$ outputting samples from $\mathcal{D}_b$. The DMSIDH problem is, given access to $\mathcal{O}^{\mathsf{DMSIDH}}$ and $(E_1, \phi(P_2), \phi(Q_2))$, to determine b. The problem is visualised in Figure 2.3.

In other words, $(E_1, \phi(P_2), \phi(Q_2))$ is an SIDH public key, and the ψ and ψ' are the vertical sides of an SIDH square. The challenge is to determine whether a point T' is the actual image of T under the hidden horizontal isogeny on the fourth (bottom) side of the SIDH square (which is guaranteed to exist).

$$\begin{tikzcd}
(E_0, P_2, Q_2) \arrow[r, dashed, "\phi"] \arrow[d, "\psi"'] & (E_1, \phi(P_2), \phi(Q_2)) \arrow[d, "\psi'"] \\
(E_2, S, T) \arrow[r, dashed, "\text{does } T' = \phi'(T)?"] & (E_3, \phi'(S), T')
\end{tikzcd}$$

Figure 2.3: The Decisional Mirror SIDH (DMSIDH) problem (Definition 2.13) visualised. Dashed lines are secret and are not known by the adversary/distinguisher. S is such that $\langle S \rangle = \ker(\widehat{\psi})$.

Observe that, given an SIDH public key, one can already choose isogenies ψ, ψ' such that $\ker(\psi') = \phi(\ker(\psi))$. We can also obtain a point S and its image $\phi'(S)$ via these ψ and ψ'. This is possible due to Lemma 1.8 (and achieved in practice using the algorithm in Figure 1.1) Thus, the only additional information provided in the DMSIDH problem is a candidate image T' of one extra point T on E_2 (independent of S).

The advantage of an adversary $\mathcal{A}$ against the DMSIDH problem is defined as

$$\mathsf{Adv}^{\mathrm{dmsidh}}(\mathcal{A}) = \left| \Pr\left[b = b' \;\middle|\; \begin{array}{c} ((E_1, \phi(P_2), \phi(Q_2)), \mathcal{O}) \leftarrow \mathsf{Gen}^{\mathsf{DMSIDH}}(\mathsf{pp}) \\ b' \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pp}, (E_1, \phi(P_2), \phi(Q_2))) \end{array} \right] - \frac{1}{2} \right|, \tag{2.14}$$

where b is the secret bit generated by $\mathsf{Gen}^{\mathsf{DMSIDH}}$, and the probability is taken over the random coins used by $\mathsf{Gen}^{\mathsf{DMSIDH}}$.

2.3.1 Double variants

In Section 4.4, we propose a new sigma protocol that uses two independent SIDH squares simultaneously. In order to prove zero-knowledge of this scheme, we define a "double" variant of the DSSP problem. For clarity of the same proof, we also define a double version of our new DMSIDH problem and prove that this Double-DMSIDH problem is hard if the "single" version is. Hence, this latter definition is only needed as a tool to simplify the security proof.

Recall from Definition 1.6 the precise meaning of independence here. By slight abuse of terminology, we say two SIDH squares are independent if both use the same isogeny ϕ for Alice, but independent isogenies ψ_0, ψ_1 for Bob, in the sense of Definition 1.6.

The Double-DSSP problem differs from the "single" version by the introduction of two bases U_i', V_i' of the $\ell_1^{e_1}$-torsion subgroups on $E_{2,i}$, for $i \in \{0, 1\}$. As we shall see in Chapter 4, these extra points will be used to verify that the two independent SIDH squares in the "double" protocol both use consistent isogenies ϕ_i' (that is, both arise from the same hidden ϕ in the two SIDH squares). These extra points, plus the requirement that the isogenies ψ_i used in each of the two squares should be independent, mean a reduction from DSSP to the Double-DSSP problem is unlikely. We believe Double-DSSP is a hard problem.

Definition 2.14 (Double-DSSP Problem)**.** For public parameters $(E_0, P_2, Q_2, E_1, \phi(P_2), \phi(Q_2))$, let $\mathcal{O}^{\mathsf{DSSP}}$ be a DSSP instance generator oracle (with secret bit b). The Double-DSSP problem is to distinguish between the following two distributions:

- $\mathcal{D}_0 = \{(\mathrm{inst}_i, U'_i, V'_i)_{i\in\{0,1\}}\}$ where $\mathrm{inst}_i = (E_{2,i}, E_{3,i}, \phi'_i) \leftarrow \mathcal{O}^{\mathsf{DSSP}}$ with $b = 0$, and additionally, if $\psi_i : E_0 \to E_{2,i}$ are the respective isogenies of degree $\ell_2^{e_2}$, then ψ_0 and ψ_1 are independent and $U'_i, V'_i = \psi_i(U), \psi_i(V)$ for $\{U, V\}$ a random basis of $E_0[\ell_1^{e_1}]$.
- $\mathcal{D}_1 = \{(\mathrm{inst}_i, U'_i, V'_i)_{i\in\{0,1\}}\}$ where $\mathrm{inst}_i = (E_{2,i}, E_{3,i}, \phi'_i) \leftarrow \mathcal{O}^{\mathsf{DSSP}}$, $b = 1$, U'_i, V'_i is a random basis of the $\ell_1^{e_1}$-torsion subgroup on $E_{2,i}$ such that $e_{\ell_1^{e_1}}(U'_0, V'_0) = e_{\ell_1^{e_1}}(U'_1, V'_1)$, and there is a pair (a, b) of integers such that the kernel of ϕ'_i is generated by $[a]U'_i + [b]V'_i$ for both $i \in \{0, 1\}$.

Intuitively, the DSSP problem states that a distinguisher cannot even determine whether an isogeny of a certain degree exists between the individual pairs of elliptic curves. This double problem only introduces the extra condition that *if* the isogenies exist, they are independent, and the preimages of the points U'_i, V'_i under these isogenies are equal. The DSIDH problem from Definition 2.3 already assumes that it is hard to decide whether a coprime torsion basis is actually the image of another under a hidden isogeny, and we cannot imagine a scenario in which one could determine whether the isogenies are independent or not without even knowing they exist. Thus, we believe this is a reasonable assumption.

Definition 2.15 (Double-DMSIDH Problem)**.** For public parameters $\mathsf{pp} = (E_0, P_2, Q_2)$, let

$$(E_1, \phi(P_2), \phi(Q_2)), \mathcal{O}^{\mathsf{DMSIDH}} \leftarrow \mathsf{Gen}^{\mathsf{DMSIDH}}(\mathsf{pp})$$

be a DMSIDH instance with secret bit b. Define the distribution $\mathcal{D}_b = \{(\mathrm{inst}_i)_{i\in\{0,1\}}\}$ where $\mathrm{inst}_i = (\psi_i, \psi'_i, S_i, T_i, \phi'_i(S_i), T'_i) \leftarrow \mathcal{O}^{\mathsf{DMSIDH}}$, and ψ_0 and ψ_1 are independent. The Double-DMSIDH problem is, given access to an oracle that samples from $\mathcal{D}_b$, to determine b.

Theorem 2.16. *If there exists an adversary $\mathcal{A}$ which makes n queries to a Double-DMSIDH oracle and guesses its bit b with advantage* Adv*, then there exists an adversary that solves the DMSIDH problem (with oracle $\mathcal{O}^{\mathsf{DMSIDH}}$) with the same advantage* Adv*, after making an expected $n(\ell_2 + 1)/\ell_2$ queries to $\mathcal{O}^{\mathsf{DMSIDH}}$.*

Proof. The adversary $\mathcal{A}$ is given public parameters pp and public key $(E_1, \phi(P_2), \phi(Q_2))$, and has access to a Double-DMSIDH oracle $\mathcal{O}$ in an analogous manner to Equation 2.14. The reduction takes as input a DMSIDH instance $((E_1, \phi(P_2), \phi(Q_2)), \mathcal{O}^{\mathsf{DMSIDH}})$ and executes $\mathcal{A}$ on input $(E_1, \phi(P_2), \phi(Q_2))$. When $\mathcal{A}$ makes a Double-DMSIDH oracle query we respond as follows: we first query $\mathcal{O}^{\mathsf{DMSIDH}}$ for $\mathrm{inst}_0 = (\psi_0, \psi'_0, S_0, T_0, \phi'_0(S_0), T'_0)$, then we keep querying $\mathcal{O}^{\mathsf{DMSIDH}}$ for $\mathrm{inst}_1 = (\psi_1, \psi'_1, S_1, T_1, \phi'_1(S_1), T'_1)$ until ψ_0 and ψ_1 are independent. Finally, we return $(\mathrm{inst}_0, \mathrm{inst}_1)$.

Write $\ker(\psi_i) = \langle [a_i]P_2 + [b_i]Q_2\rangle$, and say that two pairs a_i, b_i $(i \in \{0, 1\})$ are conjugate if $(a_0, b_0) = \lambda(a_1, b_1)$ for some invertible scalar λ. There are $\ell_2 + 1$ different such conjugacy classes of (a_i, b_i), and being in different conjugacy classes implies that $a'_0 b'_1 - a'_1 b'_0$ is invertible. Thus, with probability $\ell_2/(\ell_2 + 1)$, any two random choices of ψ_i will be independent.

Therefore, if $\mathcal{A}$ makes n queries to the Double-DMSIDH oracle, the simulation makes an expected number $n(\ell_2+1)/\ell_2$ of queries to $\mathcal{O}^{\mathsf{DMSIDH}}$. Because the simulation is perfect, whatever advantage

$\mathcal{A}$ has against Double-DMSIDH carries over to DMSIDH. □

2.3.2 SI-CDH-based assumptions

We now present two new computational assumptions, both based on the standard SI-CDH problem from Definition 2.6. We sketch proofs that the SI-CDH problem can be reduced to both, in the random oracle model. These two assumptions are simply tools used in Chapter 5 to simplify the proof of security of our new post-quantum Signal key agreement protocol, called SI-X3DH.

As usual, let pp be fixed SIDH public parameters. For ease of notation, let $\mathcal{K}_i$ (the i-keyspace) be the set of possible isogenies of degree $\ell_i^{e_i}$ from the fixed SIDH base curve E_0. Equivalently, $\mathcal{K}_i$ is the set of points of exact order $\ell_i^{e_i}$ on E_0, acting as isogeny kernel generators, where two generators are identified as the same key if they generate the same kernel.

Let $H_1 : \{0,1\}^* \to \mathcal{K}_i$ be a pseudorandom generator (PRG) whose codomain is this secret isogeny keyspace. We also let $H_2 : \{0,1\}^* \to \{0,1\}^\kappa$ be a PRG. Both H_1 and H_2 are modelled as random oracles. $\mathsf{PubkeyFromSecret}$ is a function taking a secret isogeny or kernel generator and outputting the usual SIDH public key tuple corresponding to that isogeny (or the isogeny with that kernel, via Vélu's formulae).

The first new SI-CDH-type assumption we define adds a "check" oracle to the SI-CDH assumption, which is provided by the challenge generator. This lets the adversary "verify" their answer before returning it to the challenger, so we call this the **Verifiable SI-CDH problem**.

Definition 2.17 (Verifiable SI-CDH (VCDH) problem)**.** Let pp be SIDH public parameters, and

$$K_1 = (E_1, \phi_1(P_2), \phi_1(Q_2)),$$
$$K_2 = (E_2, \phi_2(P_1), \phi_2(Q_1)),$$

be two SIDH public keys, where $\phi_i : E_0 \to E_i$ has degree $\ell_i^{e_i}$.

Let $\mathcal{O}_{K_1,K_2}$ be a truth oracle defined as

$$\mathcal{O}_{K_1,K_2}(j') = \begin{cases} 1 & \text{if } j' = \mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2), \\ 0 & \text{otherwise.} \end{cases}$$

The Verifiable SI-CDH problem is, given pp, K_1, K_2, and $\mathcal{O}_{K_1,K_2}$, to compute the j-invariant $j = \mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2)$.

Essentially oracle $\mathcal{O}_{K_1,K_2}$ confirms if the answer to the SI-CDH challenge is correct or not. Ergo, intuitively we should learn no extra information from this oracle—on all except one j-invariant the oracle will return 0, so in polynomially-many queries, the likelihood of guessing the correct j-invariant is negligible (as in the SI-CDH problem). We emphasise that K_1 and K_2 are fixed into the definition of the oracle, making the oracle significantly less powerful than a DDH oracle that accepts keys as input. We show that, in the random oracle model, this problem is hard if the SI-CDH problem is.

Theorem 2.18. *Let $\mathcal{B}$ be an adversary solving the VCDH problem with advantage ϵ after making q queries to the oracle $\mathcal{O}_{K_1,K_2}$. Then $\mathcal{B}$ can be used to solve the SI-CDH problem with probability at least $\epsilon/2q$.*

Proof. Without loss of generality, we assume all q queries are made with distinct inputs. Let (K_1, K_2) be an SI-CDH challenge instance. We define two different oracles $\mathcal{O}^0$ and $\mathcal{O}^1$. Oracle $\mathcal{O}^0$ will return 0 regardless of the query made. To define oracle $\mathcal{O}^1$, we select a random index $0 \leq \ell < q$ and let $\mathcal{O}^1$ return 1 on the ℓ-th unique query (and 0 on all other queries). We run the adversary $\mathcal{B}$ in two settings, giving instance $(K_1, K_2, \mathcal{O}^i)$ to $\mathcal{B}$ in setting $i \in \{0, 1\}$. Define found to be the event that $\mathcal{B}$ makes a query to the oracle $\mathcal{O}$ it is given with the correct j-invariant (the solution to the SI-CDH instance). We can consider the probability of $\mathcal{B}$ succeeding against the VCDH problem as

$$\begin{aligned}\epsilon = &\Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ occurs}\,] \cdot \Pr[\,\mathsf{found} \text{ occurs}\,] \\ &+ \Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ does not occur}\,] \cdot \Pr[\,\mathsf{found} \text{ does not occur}\,].\end{aligned}$$

If found does not occur, then $\mathcal{B}$ running in setting 0 (where oracle $\mathcal{O}^0$ always returns 0) will be unable to distinguish the simulated oracle from the true one, and will win with advantage ϵ. Hence,

$$\Pr[\,\mathcal{B} \text{ wins in setting } 0\,] \geq \Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ does not occur}\,].$$

On the other hand, if found occurs, then we correctly simulated the oracle in setting 1 with probability $1/q$ (the probability that we guessed ℓ correctly). Therefore,

$$\Pr[\,\mathcal{B} \text{ wins in setting } 1\,] \geq \frac{1}{q} \Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ occurs}\,].$$

We uniformly sample $b \leftarrow \{0, 1\}$ and return the solution from $\mathcal{B}$ running in setting b to the SI-CDH challenger. Because $0 \leq \Pr[\,\mathsf{found} \text{ occurs}\,] \leq 1$, we solve the SI-CDH instance with overall probability

$$\begin{aligned}&\frac{1}{2}\Pr[\,\mathcal{B} \text{ wins in setting } 0\,] + \frac{1}{2}\Pr[\,\mathcal{B} \text{ wins in setting } 1\,] \\ \geq\ &\frac{1}{2}\Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ does not occur}\,] + \frac{1}{2q}\Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ occurs}\,] \\ \geq\ &\frac{1}{2q}\left(\Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ does not occur}\,] + \Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ occurs}\,]\right) \\ &\qquad \geq \frac{1}{2q}\epsilon,\end{aligned}$$

which is non-negligible if ϵ is (since q must be polynomially-sized). □

We call the second of our new SI-CDH-type problems the **Honest SI-CDH problem** (HCDH). This problem models an SI-CDH instance with an additional FO-like proof that the first key in the instance, K_1, was honestly generated.

Definition 2.19 (Honest SI-CDH (HCDH) problem)**.** Let pp be SIDH public parameters, and $s \leftarrow \{0, 1\}^\kappa$ be a random seed, where κ is the security parameter. Then, let

$$K_1 = \mathsf{PubkeyFromSecret}(H_1(s))$$

be a public key derived from s, where $H_1(s)$ is an isogeny of degree $\ell_1^{e_1}$. Let

$$K_2 = (E_2, \phi_2(P_1), \phi_2(Q_1))$$

be a second public key, where $\phi_2 : E_0 \to E_2$ has degree $\ell_2^{e_2}$. Finally, let π be an FO-proof of the form

$$\pi = s \oplus H_2(\mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2)).$$

The Verifiable SI-CDH problem is, given pp, K_1, K_2, and π, to compute the j-invariant $j = \mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2)$.

We argue that the FO-like proof leaks no information because we obviously assume that $\mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2)$ is unknown (since it is the answer to the SI-CDH problem) and s is random. Thus, if the SI-CDH problem is hard, then so too is this problem. We sketch a reduction in the random oracle model. Treat H_2 as a random oracle. Let $\mathcal{B}$ be an adversary making q queries to H_2 and winning with advantage ϵ against the HCDH problem. Obtain an SI-CDH challenge (K_1, K_2). Choose π to be a random binary string, and provide (K_1, K_2, π) to $\mathcal{B}$.

In order to distinguish the simulated π from an honest FO-proof, $\mathcal{B}$ must query $H_2(j)$ for the correct j-invariant solution of the SI-CDH instance. If this occurs, we can return one of the q queries made to H_2 and win with probability $1/q$. Otherwise, the output of $\mathcal{B}$ wins with advantage ϵ despite π being uniformly random, by a simple hybrid argument.

Thus, the reduction can simply return one of the q queries to H_2 or the output of $\mathcal{B}$ to the SI-CDH challenger with equal probability. In either case, there is a non-negligible chance that the returned value wins the SI-CDH challenge, if ϵ is non-negligible.

Chapter 3

Adaptive Attacks and Public Key Validation

In this chapter, we delve into the adaptive attacks that the SIDH key exchange protocol is known to be vulnerable to. These attacks have had a significant impact on the adoption of SIDH as a post-quantum key exchange mechanism. For example, in the context of internet security layers (SSL/TLS), web servers will generally have a fixed or static key covered by a certificate. Using SIDH naively in this context would be insecure because the server's fixed secret could be progressively learned by malicious clients. Known solutions to this insecurity add varying degrees of complexity and computational cost to the exchange.

We begin by recapitulating the original adaptive attack on SIDH by Galbraith, Petit, Shani, and Ti [GPST16]—known as the GPST attack in reference to the authors. Following this, we examine the most notable countermeasures to this attack proposed in the literature. In short, there are two primary directions such countermeasures can take—making the GPST attack infeasible by running many parallel instances of SIDH (as k-SIDH does), or validating the public keys to ensure they are not maliciously formed (as Kirkwood et al. [KLM$^+$15] propose, using Fujisaki–Okamoto-like transformations [HHK17]). Throughout the chapter, we implicitly assume the notation of SIDH from Section 1.3.

We note that, while the GPST attack is the first adaptive attack on SIDH, it is not the only one. Other adaptive attacks have been proposed since, both on SIDH and on related schemes. For example, Fouotsa and Petit [FP22] present another attack on SIDH-type schemes based on torsion-point attacks. Ueno et al. [UXT$^+$22] show how the GPST attack can be used in the specific instance of side-channel attacks on SIKE. All such attacks have effectively the same impact, so we use the GPST attack as the prototypical example in our discussion.

3.1 The GPST attack on SIDH

The premise of the GPST attack is that a secret ℓ^e-isogeny path can be progressively learned one ℓ-isogeny step at a time. In the case of $\ell = 2$, this corresponds directly to learning the secret key α bit-by-bit. We shall demonstrate this case here, with the implicit understanding that the attack can be generalised to other primes if desired.

The attack is presented in a security model where the adversary is challenged to learn a secret key α, and has access to an oracle defined as follows:

$$\mathcal{O}(E, R, S, E') = \begin{cases} 1 & \text{if } j(E') = j(E/\langle R + [\alpha]S), \\ 0 & \text{otherwise,} \end{cases} \tag{3.1}$$

where $R, S \in E$.

Such an oracle is feasible in real-world settings, for example where participants verify whether the secrets they generated in an exchange match via a MAC or other form of symmetric-key authentication.

Let Bob be the attacker, attempting to learn Alice's static secret key α. Let Alice's public key be (E_A, P'_A, Q'_A) and suppose Bob has a secret isogeny ϕ_B, with codomain E_B. Bob will begin by computing the shared secret curve E_{AB} using ϕ_B and Alice's public key, as in the SIDH protocol.

Let α_i denote the $(i+1)$-th bit of α, where α_0 is the least significant bit. We define the ***i*-th partial key** K_i of a key α as

$$K_i = \sum_{k=0}^{i-1} \alpha_k 2^k. \tag{3.2}$$

Thus, the key α can be written in the form $\alpha = K_i + \alpha_i 2^i + \alpha' 2^{i+1}$ for some α'.

Suppose Bob has learned the first i bits of α, and wishes to learn the $(i+1)$-th bit. This includes the case where $i = 0$ and $K_i = 0$. Note that the original paper [GPST16] takes more care to distinguish between the general cases of equivalent keys (see Definition 1.4), but we take the simpler, less general route here and assume keys of the form $(1, \alpha)$. Bob will compute

$$\begin{aligned} R_i &= R + [-2^{n-i-1}K_i]S, \\ S_i &= [1 + 2^{n-i-1}]S, \end{aligned} \tag{3.3}$$

where $R = \phi_B(P_1)$ and $S = \phi_B(Q_1)$. His curve E_B together with these crafted points R_i, S_i will form the public key Bob gives to Alice.

Now, after Alice has been given the key (E_B, R_i, S_i) from Bob, she will compute

$$\begin{aligned} &R_i + [\alpha]S_i \\ &= R + [-2^{n-i-1}K_i]S + [\alpha][1 + 2^{n-i-1}]S \\ &= R + [\alpha]S + [2^{n-i-1}\alpha - 2^{n-i-1}K_i]S \\ &= R + [\alpha]S + [2^{n-1}\alpha_i]S \\ &= \begin{cases} R + [\alpha]S & \text{if } \alpha_i = 0, \\ R + [\alpha]S + [2^{n-1}]S & \text{if } \alpha_i = 1. \end{cases} \end{aligned} \tag{3.4}$$

Hence, an oracle query of the form $\mathcal{O}(E_B, R_i, S_i, E_{AB})$ will return 1 if and only if $\alpha_i = 0$. Bob has consequently learned the next bit of α, and repeats this process until all of α is recovered.

The attack, therefore, only requires making a linear (in the bit-length of the secret) number of queries to the oracle $\mathcal{O}$ in order to recover the full key.

3.2 k-SIDH

Azarderakhsh, Jao, and Leonardi [AJL17] propose a natural method of defending against the GPST attack, by running multiple instances of the SIDH exchange in parallel and combining all the resulting shared secrets. The name k-SIDH designates such a protocol in which both parties use k keys each, giving rise to a total of k^2 individual SIDH squares. Specifically, Alice randomly generates k different secret keys

$$\alpha_i \leftarrow \mathbb{Z}/2^n\mathbb{Z} \qquad 0 \le i < k,$$

and then computes the k corresponding public key tuples $(E_{A,i}, P_{A,i}, Q_{A,i})$, where

$$\phi_{A,i} \ : \ E_0 \to E_{A,i} \text{ has kernel } \langle P_1 + [\alpha_i]Q_1 \rangle, \tag{3.5}$$

and $P_{A,i}, Q_{A,i} = \phi_{A,i}(P_2), \phi_{A,i}(Q_2)$.

Bob, likewise, generates secrets

$$\beta_i \leftarrow \mathbb{Z}/3^n\mathbb{Z} \qquad 0 \le i < k,$$

and computes his k corresponding public key tuples $(E_{B,j}, P_{B,j}, Q_{B,j})$ from isogenies

$$\phi_{B,j} \ : \ E_0 \to E_{Bj} \text{ with kernel } \langle P_2 + [\beta_j]Q_2 \rangle, \tag{3.6}$$

with $P_{B,j}, Q_{B,j} = \phi_{B,j}(P_1), \phi_{B,j}(Q_1)$.

Alice's public key for the k-SIDH exchange is $\big((E_{A,i}, P_{A,i}, Q_{A,i})_{i \in \{0,\dots,k-1\}}\big)$, and Bob's is similarly $\big((E_{B,j}, P_{B,j}, Q_{B,j})_{j \in \{0,\dots,k-1\}}\big)$. After exchanging public keys, Alice will compute k^2 different SIDH key exchanges—one for each pair of tuples from her public key and Bob's public key. Denote the shared secret from each of these exchanges by

$$z_{i,j} = j\left(E_{B,j}/\langle P_{B,j} + [\alpha_i]Q_{B,j}\rangle\right) \qquad 0 \le i, j < k. \tag{3.7}$$

Alice can then combine all k^2 of these values as follows, using H, a preimage resistant hash function:

$$h = H(z_{0,0} \parallel \dots \parallel z_{0,k-1} \parallel \dots \parallel z_{k-1,0} \parallel \dots \parallel z_{k-1,k-1}). \tag{3.8}$$

In the same manner, Bob will also compute k^2 shared secrets, and combine them in an identical way.

$$\begin{aligned} z'_{i,j} &= j\left(E_{A,i}/\langle P_{A,i} + [\beta_j]Q_{A,i}\rangle\right) \qquad 0 \le i, j < k, \\ h' &= H(z'_{0,0} \parallel \dots \parallel z'_{0,k-1} \parallel \dots \parallel z'_{k-1,0} \parallel \dots \parallel z'_{k-1,k-1}). \end{aligned} \tag{3.9}$$

Alice and Bob then confirm they have both derived the same shared key, $h = h'$.

The intuition behind this scheme is that with the many interactions between all of Bob's keys and all of Alice's, there are too many "moving parts" for a malicious Bob to feasibly compute the secret bits of Alice's k keys. Any manipulation of a single SIDH tuple in Bob's public key will affect at least k of the $z_{i,j}$ values from which the shared secret is computed, and Bob can only ever derive a matching secret key if all k least significant bits of the secrets are guessed simultaneously. Otherwise, even a single incorrect guess for one of the $z_{i,j}$ values will cause the hash h' to be different from the one Alice derives.

At first glance, one may wonder if even $k = 2$ is sufficient to thwart the GPST attack—both of Bob's keys are used with both of Alice's to derive the shared secret, so the complexity of the attack is much higher than in the plain SIDH setting. The hashing of four different derived secrets may be enough to hide any information about the individual exchanges involved. Unfortunately, Dobson, Galbraith, LeGrow, Ti, and Zobernig [DGL$^+$20] demonstrate a practical extension of the GPST attack to k-SIDH for small k, for example $k = 2$, the case covered explicitly by the authors. The DGLTZ attack scales exponentially with k. The additional complexity of the attack, compared to GPST, is due to the necessity of recovering intermediate curves and certain points on them by enumerating possible neighbouring curves at each step. The k-SIDH scheme is, therefore, confirmed to be insecure for small k.

Instead, Azarderakhsh et al. [AJL17] propose the use of $k = 60$ (resp. 50) if the static-key user is working in the 2^n-torsion (resp. 3^m) subgroup for 128-bits of classical security, and $k = 113$ (resp. 94) to achieve 128-bits of quantum security. For 128-bit quantum security, therefore, the protocol concretely requires 10,622 individual SIDH exchanges to be computed. This is, unfortunately, very slow—on top of the key sizes, which are a hundred times larger than in SIDH. Because using isogenies of larger degree reduces the required number of instances k, it has been noted that larger primes may be more efficient overall than using $\ell_1 = 2$, $\ell_2 = 3$, despite each individual isogeny computation being slower.

Urbanik and Jao [UJ20] presented an attempt at improving the efficiency of k-SIDH by using the non-trivial automorphisms on curves with j-invariant 0 and 1728 to derive extra secrets. As briefly discussed in Section 1.2, these curves have multiple edges leaving them with the same codomain curve. Let's consider E_0, where $j(E_0) = 0$. For a single isogeny $\phi : E' \to E_0$, there will be three non-equivalent isogenies $\tilde{\phi}_i : E_0 \to E'$, each of which is dual to an isogeny equivalent to ϕ. Hence, if Alice takes a secret isogeny $\phi_{A,0}$ with kernel G_A on E_0, she can also obtain two other secret isogenies to the same curve by acting on G_A with η_6:

$$\begin{aligned} \phi_{A,0} : E_0 \to E_A &\qquad \text{with kernel } G_A, \\ \phi_{A,1} : E_0 \to E_A &\qquad \text{with kernel } \eta_6(G_A), \\ \phi_{A,2} : E_0 \to E_A &\qquad \text{with kernel } \eta_6^2(G_A). \end{aligned} \tag{3.10}$$

These three isogenies will, in general, all be distinct in the first "step" in their paths, when considered as compositions of prime-degree isogeny steps. Thus, they are non-equivalent as isogenies, since their kernels differ.

Bob can take the same approach to obtain three isogenies from a single secret kernel G_B on E_0 as well. Performing nine SIDH exchanges with all combinations of the three keys from each party results in nine j-invariants. However, only three of these are distinct, since

$$\begin{aligned} E_0/\langle G_A, G_B\rangle &\cong E_0/\langle \eta_6(G_A), \eta_6(G_B)\rangle &&\cong E_0/\langle \eta_6^2(G_A), \eta_6^2(G_B)\rangle, \\ E_0/\langle G_A, \eta_6(G_B)\rangle &\cong E_0/\langle \eta_6(G_A), \eta_6^2(G_B)\rangle &&\cong E_0/\langle \eta_6^2(G_A), G_B\rangle, \\ E_0/\langle \eta_6(G_A), G_B\rangle &\cong E_0/\langle \eta_6^2(G_A), \eta_6(G_B)\rangle &&\cong E_0/\langle G_A, \eta_6^2(G_B)\rangle. \end{aligned} \tag{3.11}$$

Urbanik and Jao [UJ20] argue that if both parties use triples of secrets derived in this way, then a smaller value of k can be used in k-SIDH, improving efficiency and lowering the size of the public keys.

Unfortunately, Basso, Kutas, Merz, Petit, and Weitkämper [BKM+20] demonstrate an adaptive attack on this scheme which is more efficient than the one on k-SIDH, exploiting the extra structure of the secrets. They consequently show that the Urbanik and Jao scheme actually scales worse in efficiency than k-SIDH itself, requiring around twice as many SIDH instances for the same security. It does still result in slightly smaller public keys, though—the public keys are around 4/5 of the size of those in k-SIDH. We also emphasise that these ideas cannot be used when the base SIDH curve is not E_0 or E_{1728}.

Remark 3.1. Urbanik and Jao [UJ20] also suggest a non-interactive scheme for proving correctness of an SIDH public key (including the torsion points), using the same idea of employing the non-trivial automorphisms on curves E_0 and E_{1728}, just as their k-SIDH improvement proposal does. The idea is that the ephemeral isogeny used in each round of the scheme can be chosen such that the additional secrets from the automorphism η are pairwise-independent (in the same sense as Definition 1.6), and that this prevents GPST-style adaptive attacks. We show in Chapter 4 that the soundness proof of this protocol is incorrect, so we do not cover it in detail here. We refer the interested reader to their work [UJ20, Section 5].

3.3 The Weil pairing check

We have already seen, in Chapter 2, that the Weil pairing can be used to provide additional validation of points in SIDH public keys. We shall now provide more detail about this pairing and its use in validation. An excellent reference for the material covered here is given by Silverman [Sil09, Section III.8].

Suppose E is an elliptic curve defined over a field k, and let $m \geq 2$ be an integer coprime to the characteristic of k. The m-th Weil pairing on E is a map

$$e_m : E[m] \times E[m] \longrightarrow \boldsymbol{\mu}_m,$$

where $\boldsymbol{\mu}_m$ is the group of the m-th roots of unity, and $E[m]$ is the m-torsion subgroup of E (as before).

The actual definition of the pairing requires knowledge of divisors, which we introduce in Section 6.4, and understanding its definition is not critical in this thesis—we can simply take its existence for granted. In short, it can be defined as

$$e_m(S,T) = \frac{g(S+X)}{g(X)}, \tag{3.12}$$

where g is a function with divisor

$$(g) = \sum_{R \in E[m]} (T' + R) - (R), \tag{3.13}$$

for any point $T' \in E$ such that $[m]T' = T$, and where $X \in E$ is any point such that both numerator and denominator of e_m are defined and non-zero. Note that $T' + R$ is point addition on the curve, but the sum is a formal sum of points. It happens that this definition is independent of the choice of T', X, and g. The interested reader can refer to Silverman [Sil09] for more detail and proofs. We are primarily interested in the properties this pairing e_m provides us with, so we will examine some of these now.

In particular, e_m is bilinear, so that

- $e_m(S_1 + S_2, T) = e_m(S_1, T) \cdot e_m(S_2, T)$,
- $e_m(S, T_1 + T_2) = e_m(S, T_1) \cdot e_m(S, T_2)$.

It is also alternating, implying

- $e_m(S,S) = 1$,
- $e_m(S,T) = e_m(T,S)^{-1}$.

Finally, it is non-degenerate, so that if $e_m(S, T) = 1$ for all $T \in E[m]$, then $S = \mathcal{O}_E$ (and vice versa).

We now turn to the use of this pairing in validation of SIDH public keys. Let $\phi : E \to E'$ be an isogeny, and let $P, Q \in E$ be points of order N, where N is coprime to $\deg(\phi)$. Following from Silverman [Sil09, Proposition III.8.2], the Weil pairing induces the condition that

$$e_N(\phi(P), \phi(Q)) = e_N(P, Q)^{\deg(\phi)}, \tag{3.14}$$

where the first pairing is computed on E, and the second on E'.

Consider an SIDH public key of the form (E', P', Q'). While we do not know the secret isogeny ϕ corresponding to this key, we *do* know its degree, which is part of the SIDH public parameters. So we can compute $e_N(P, Q)^{\deg(\phi)}$, and we can compute $e_N(P', Q')$, and test for equality. This was discussed by Galbraith et al. [GPST16, GV18] as a potential method of validating SIDH public keys to protect against adaptive attacks such as the GPST attack. Unfortunately, this pairing check is insufficient to prevent such attacks. The check holds for many different choices of basis points, hence is not enough to uniquely determine whether a secret isogeny ϕ of correct degree exists in SIDH.

For example, let us return to our usual choice of $N = \ell_2^{e_2}$ and $\deg(\phi) = \ell_1^{e_1}$. We note that there are

$$\ell_2^{4e_2-3} \cdot (\ell_2^2 - 1)^2 / (\ell_2 + 1) \tag{3.15}$$

different possible choices for ordered bases P, Q of the correct order—this is because there are $\ell_2^{2e_2} - \ell_2^{2(e_2-1)}$ points of the correct order, and the independence between P and Q introduces a factor of $\ell_2/(\ell_2 + 1)$. Obviously, many of these choices will not satisfy the Weil pairing check for a certain degree. However, the codomain of e_N has order $\ell_2^{e_2}$, which is much smaller than the number of choices of points. Therefore there will be far more pairs of points satisfying the Weil pairing check, than could possibly arise from valid secret isogenies of the correct degree.

Indeed, if (P', Q') are the correct images of (P, Q) under an isogeny ϕ, then any pair $([a]P' + [b]Q', [c]P' + [d]Q')$ such that $ad - bc \equiv 1 \pmod{N}$ also passes the Weil pairing check. This can be seen via the bilinearity of e_N. Hence, this is not enough to uniquely determine ϕ, and, in particular, is insufficient to protect against the GPST adaptive attack. As shown by Galbraith et al. [GPST16], it is simple to circumvent the Weil pairing check in the GPST attack by scaling the points $[\theta]R_i, [\theta]S_i$ (where R_i, S_i are the maliciously chosen points seen in Section 3.1) using an appropriately chosen θ (coprime to N, the order of the points).

3.4 The Fujisaki–Okamoto transformation

The Fujisaki–Okamoto transformation [FO99, FO13] is a generic transformation from any one-way (OW-CPA) secure asymmetric encryption scheme, and any one-time secure symmetric encryption scheme, into an IND-CCA (indistinguishable under chosen-ciphertext attacks) secure hybrid encryption scheme in the random oracle model. Specifically, the hybrid encryption is defined as

$$\begin{aligned} \mathrm{Enc}_{pk}^{\mathsf{hy}}(m; r) &:= \mathrm{Enc}_{pk}^{\mathsf{asym}}(r; H(r, c)) \parallel c \\ \text{where } c &\leftarrow \mathrm{Enc}_{G(r)}^{\mathsf{sym}}(m). \end{aligned} \tag{3.16}$$

Here, G, H are hash functions (modelled as random oracles), r denotes the random coins used by $\mathrm{Enc}_{pk}^{\mathsf{hy}}$ (assumed to be chosen at random from an appropriate domain), $\mathrm{Enc}_{pk}^{\mathsf{asym}}(m; c')$ is a

OW-CPA secure asymmetric encryption scheme with public key pk on message m using random coins c', and $\mathrm{Enc}_k^{\mathsf{sym}}$ is a one-time secure symmetric encryption algorithm with key k.

One-wayness is a very weak security definition for encryption, which only asserts that an adversary cannot entirely decrypt the encryption of a random plaintext. Intuitively, this means r cannot be (entirely) recovered from $\mathrm{Enc}_{pk}^{\mathsf{asym}}(r; H(r, c))$ by an adversary. Hence, because $G(r)$ is hidden from the adversary and only used once (since r is random), and because $\mathrm{Enc}^{\mathsf{sym}}$ is one-time secure, we can achieve an IND-CCA secure scheme $\mathrm{Enc}^{\mathsf{hy}}$ generically.

Hofheinz, Hövelmanns, and Kiltz [HHK17] give more detailed analysis and generalisations of this transformation. We use "FO transformation" to loosely refer to all transformations in this vein. The FO transformation has found a wide variety of uses, and in this thesis we are specifically interested in its application to key exchange protocols. Peikert [Pei14] and Kirkwood et al. [KLM$^+$15] have both set out methods of using the FO transformation to convert a key exchange protocol into a secure key encapsulation mechanism (KEM).

This is exactly what was done in the SIKE [ACC$^+$17] protocol, a NIST post-quantum standardisation candidate, derived from the SIDH protocol. SIKE is a secure KEM, preventing the adaptive attacks that SIDH suffers from by validating key well-formedness with the FO transformation. It was pointed out by Galbraith et al. [GPST16] that the Kirkwood et al. [KLM$^+$15] validation method using the FO transformation is sufficient for this purpose. The idea of the FO transformation is that the initiator, A, of the key exchange can encrypt the randomness they used in the exchange (for example, to generate their secret key) under the symmetric shared key K they derived, and send it to their partner B. If the encryption method is one-time secure, then because only A and B know K, only they can decrypt this randomness. B can then check that the public key A provided is indeed derived from the randomness A sent, to prove that the public key is well-formed. This allows B to detect malicious modification of the public key used by A. Clearly though, because B learns the secret key of such a public key in every exchange, A can only prove the legitimacy of *ephemeral* public keys in this way—otherwise, every exchange partner would be given A's long-term secret. Hence, while extremely useful, the FO transformation does not provide a solution in cases where parties wish to perform an exchange while both using static keys.

Concretely, using SIKE as an example, let m be a uniformly random bit-string, and suppose we are performing encapsulation under a fixed public key $pk = (E, P, Q)$. We can derive an ephemeral secret key s deterministically from m and pk (for example, by hashing: $s := H(m \| pk)$). Let c_0 be the public key corresponding to s. An SIDH key exchange between pk and c_0 gives us a shared secret r. Then c_0 can be considered the ciphertext of an asymmetric-key "encryption" of r under pk (using randomness s). The symmetric encryption used by SIKE is simply the one-time pad (OTP), so we compute $c_1 := m \oplus G(r)$.

Given c_0 and c_1, the owner of pk can "decrypt" c_0 by recomputing the shared secret $r \leftarrow \mathrm{SIDH}(pk, ek)$ (the asymmetric step). After this, they can recover m as $G(r) \oplus c$. The final step in validation (outside the scope of the FO transformation itself) is to confirm that this recovered randomness m does indeed produce s (and c_1)—convincing the verifier that c_1 is honestly generated and that the exchange with pk could not have involved an adaptive attack. This allows the verifier to trust the shared secret produced, but leaks the secret key s, so c_1 cannot be securely reused by the sender.

Chapter 4

SIDH Proof of Knowledge

In Section 1.3, we reviewed the Supersingular Isogeny Diffie–Hellman (SIDH) [JD11, DJP14] protocol. While this scheme is a fast and efficient post-quantum key exchange candidate, it has been hampered by the existence of the practical adaptive attacks on the scheme discussed in Chapter 3—the first of these given by Galbraith et al. [GPST16] (the GPST attack). These attacks mean it is not safe to reuse a static key across multiple SIDH exchanges without other forms of protection. As such, various countermeasures have been proposed—though each with its unique drawbacks.

We saw in Section 3.4 that one method of protection is to require one participant to use a one-time ephemeral key in the exchange, with a Fujisaki–Okamoto-type transformation [HHK17] revealing the corresponding secret to the other party. This allows the recipient to verify the public key is well-formed, ensuring an adaptive attack was not used. This is what was done in SIKE [ACC$^+$17], and converts the scheme to a secure key encapsulation mechanism (KEM). But it is of limited use in cases where both parties wish to use a long-term key.

An alternative countermeasure, seen in Section 3.2, is to use many SIDH exchanges in parallel, combining all the resulting secrets into a single value. This scheme is known as k-SIDH [AJL17], where k is the number of keys used by each party in the exchange. The authors of the k-SIDH proposal suggest $k = 92$ is required for a secure key exchange, and it must be noted that the number of SIDH instances grows as k^2, so this scheme is very inefficient. Urbanik and Jao's [UJ20] proposal attempted to improve the efficiency of this protocol by making use of the special automorphisms on curves with j-invariant 0 or 1728, but it was shown by Basso et al. [BKM$^+$20] that Urbanik and Jao's proposal is vulnerable to a more efficient adaptive attack and actually scales worse in efficiency than k-SIDH itself (although the public keys are around 4/5 of the size, it requires around twice as many SIDH instances for the same security).

Finally, adaptive attacks can also be prevented by providing a non-interactive proof that a public key is well-formed or honestly generated. While generic NIZK proof systems would make this possible in a very inefficient manner, Urbanik and Jao [UJ20] claim a method for doing so using a similar idea to their k-SIDH improvement mentioned above. Their scheme is built on the SIDH-based identification scheme by De Feo, Jao, and Plût [DJP14].

Unfortunately, however, we show that the soundness of this original De Feo–Jao–Plût scheme is not rigorously proved—specifically that it does not reduce to the computational assumption they claim—and give a counterexample to this proof. Precisely, a dishonest prover can successfully convince a verifier to accept their proof for a public key curve E_1 even when there is no isogeny $\phi : E_0 \to E_1$ of degree $\ell_1^{e_1}$. Because this scheme (and proof) has since been used to build an undeniable signature by Jao and Soukharev [JS14], a signature scheme by Yoo, Azarderakhsh, Jalali, Jao, and Soukharev [YAJ$^+$17], and also by Galbraith, Petit, and Silva [GPS20], all of these subsequent papers suffer from the same issue. Our counterexample does not apply to Urbanik and Jao's point-validation scheme [UJ20], but their soundness proof nonetheless does not hold for the same reason (as mentioned in Remark 3.1). Explicit counterexamples to Urbanik and Jao's scheme are given by De Feo et al. [DDGZ21].

In this chapter, we examine the issue with the existing soundness proofs and propose two new SIDH-based zero-knowledge schemes to prove that SIDH public keys are well-formed, meaning that given base SIDH curve E_0 and the curve E_1 in the public key, there is an isogeny (the private key, or witness) $\phi : E_0 \to E_1$ of the correct degree.

First, in Section 4.3, we propose a modification to the De Feo–Jao–Plût scheme that ensures an extractor for the witness $\phi : E_0 \to E_1$ exists. We express this in terms of a relation we call the *weak* SIDH relation, $\mathcal{R}_{\mathsf{weakSIDH}}$. There are two key ideas used in the development of this protocol, which we briefly summarise here using the notation of the De Feo–Jao–Plût scheme, referring to Section 4.2.1 and Figure 4.1. The first idea is for the prover to provide bases (P_2, Q_2) of $E_2[\ell_2^{e_2}]$ and (P_3, Q_3) of $E_3[\ell_2^{e_2}]$, and for the verifier in the $\mathsf{chall} = 1$ case to check that $(P_3, Q_3) = (\phi'(P_2), \phi'(Q_2))$. In the $\mathsf{chall} = 0$ case, the verifier checks that the isogenies from E_2 to E_0 and from E_3 to E_1 are "parallel". The second key idea is, in the 2-special soundness proof, to view the transcript as an SIDH square where E_2 is the "base curve" and where E_0 and E_3 play the roles of E_A and E_B in SIDH. It then follows that there is a witness ϕ as required.

Second, in Section 4.4, we give a new proof that convinces a verifier not only that there is an isogeny $\phi : E_0 \to E_1$ of degree $\ell_1^{e_1}$, but also that $(P_1, Q_1) = (\phi(P_0), \phi(Q_0))$. We call this stronger relation the SIDH relation, $\mathcal{R}_{\mathsf{SIDH}}$, to distinguish it from the weak version mentioned above. Making this non-interactive using the Fiat-Shamir heuristic gives a secure method for proving well-formedness of SIDH public keys, which is needed if one wants to prevent adaptive attacks. This is the first such sound proof in the literature (since the soundness proof of Urbanik and Jao's scheme [UJ20] is invalid, as mentioned before) and has important applications in all settings where SIDH key exchange could be used with static keys. Our scheme works with any base elliptic curve, rather than being restricted to the two curves with j-invariant 0 or 1728 as in [UJ20]. While the size of our NIZK proof is larger than a k-SIDH public key of the same security level, it is much more efficient to verify than computing a k-SIDH exchange (due to the quadratic scaling of k-SIDH).

The new protocol in Section 4.4 builds on the protocol of Section 4.3 but also needs new ingredients. One key idea is that we need to ensure that the pairs of integers (a, b) used by the prover to construct their ephemeral isogenies $\psi : E_0 \to E_2$ for each commitment round of the scheme are "independent enough". To achieve this, we "double" the protocol by essentially running two sessions of the protocol from Section 4.3 for each challenge bit. The prover shows that the two instances are consistent with each other by providing images of a random torsion basis in both squares, which the verifier can check are correct. The verifier also checks that the two instances are independent (in the precise sense of Definition 1.6). This allows us to construct a 2-special soundness extractor that outputs a correct witness.

Commitments in the original De Feo–Jao–Plût scheme were formed of just two j-invariants of curves, but our new proofs require committing to various points on curves as well. This makes the proofs larger. As with the original De Feo–Jao–Plût scheme, it is non-trivial to simulate valid protocol transcripts without knowing the witness, so we only achieve computational zero-knowledge. To prove our schemes possess the computational zero-knowledge property, we rely on some of the new assumptions that we presented in Section 2.3.

In recent work, Ghantous, Pintore, and Veroni [GPV21] demonstrated that the soundness property for the De Feo–Jao–Plût scheme (and those based on it) fails for a different reason—namely, the existence of multiple isogenies of the same degree between some curves. The protocols we propose in this chapter are not vulnerable to this issue, as we briefly discuss in Remark 4.8.

This chapter begins with a revision of some standard preliminaries about sigma protocols in Section 4.1. We then recall the De Feo–Jao–Plût identification scheme in Section 4.2.1 and outline the issue with its proof of soundness (given in multiple previous works) in Section 4.2.2. Subsequently, we present the first of our new SIDH identification schemes in Section 4.3, which modifies the De Feo–Jao–Plût scheme and allows us to prove soundness (and thus security). We then show how the points in the SIDH public key can also be verified under this zero-knowledge proof scheme in Section 4.4, including some discussion about potential improvements in Section 4.4.1. We conclude the chapter with the construction of a secure signature scheme from our protocols—the first signature scheme which is a Proof of Knowledge of an SIDH secret key (including proof of torsion point correctness).

Remark 4.1. De Feo, Dobson, Galbraith, and Zobernig [DDGZ21] build on the work presented in this chapter with a protocol which satisfies the standard definition of computational zero-knowledge, without requiring any novel assumptions (for example the DMSIDH assumption). De Feo et al.'s modified protocol uses ternary challenges and has a cheating probability of 2/3. It is mentioned for completeness, but is less efficient than the protocols given in this chapter, which are more appropriate for the applications we have in mind.

4.1 Preliminaries: Sigma protocols

A sigma protocol Π_Σ for a relation $\mathcal{R} = \{(X, W)\}$ is a public-coin three-move interactive proof system consisting of two parties: a verifier V and a prover P. Recall that public-coin informally means that there are no secret sources of randomness—the verifier's coin tosses are accessible to the prover. In practice, this means the challenge sent by the verifier to the prover is uniformly random. For our purposes, a witness W can be thought of as a secret key, while the statement X is the corresponding public key. Thus, proving $(X, W) \in \mathcal{R}$ is equivalent to saying that X is a valid public key that has a corresponding secret key. We use the security parameter κ to parametrise the length of the secret keys involved.

Definition 4.2 (Sigma protocol)**.** A sigma protocol Π_Σ for a family of relations $\{\mathcal{R}\}_\kappa$ parametrised by security parameter κ consists of probabilistic polynomial-time (PPT) algorithms $((P_1, P_2), (V_1, V_2))$, where V_2 is deterministic, and we assume P_1, P_2 share states. The protocol proceeds as follows:

1. Round 1: The prover, on input $(X, W) \in \mathcal{R}$, returns a commitment $\mathsf{com} \leftarrow P_1(X, W)$ which is sent to the verifier.

2. Round 2: The verifier, on receipt of com, runs $\mathsf{chall} \leftarrow V_1(1^\kappa)$ to obtain a random challenge, and sends this to the prover.

3. Round 3: The prover then runs $\mathsf{resp} \leftarrow P_2(X, W, \mathsf{chall})$ and returns resp to the verifier.

4. Verification: The verifier runs $V_2(X, \mathsf{com}, \mathsf{chall}, \mathsf{resp})$ and outputs either $\top$ (accept) or $\bot$ (reject).

A transcript $(\mathsf{com}, \mathsf{chall}, \mathsf{resp})$ is said to be valid for a statement X if $V_2(X, \mathsf{com}, \mathsf{chall}, \mathsf{resp})$ outputs $\top$. Let $\langle P, V\rangle$ denote the transcript for interaction between prover P and verifier V, and write $\langle P, V\rangle = 1$ if the verifier accepts after the protocol is complete. We define some standard properties of sigma protocols which are relevant for this chapter:

Correctness: If the prover P knows $(X, W) \in \mathcal{R}$ and behaves honestly, then the verifier V accepts the proof.

2-special soundness: There exists a polynomial-time extraction algorithm $\mathsf{Extract}$ that, given a statement X and two valid transcripts $(\mathsf{com}, \mathsf{chall}, \mathsf{resp})$ and $(\mathsf{com}, \mathsf{chall}', \mathsf{resp}')$ where $\mathsf{chall} \neq \mathsf{chall}'$, outputs a witness W such that $(X, W) \in \mathcal{R}$ with probability at least $1 - \varepsilon$ for soundness error ε.

Zero-knowledge: We use a non-standard definition of zero-knowledge, sufficient for our applications, which we call *distributional* zero-knowledge. A sigma protocol is distributionally zero-knowledge with respect to an instance generation algorithm Gen if, for any (cheating) verifier V^*, there exists a polynomial-time simulator Sim such that the following distinguishing probability holds for all PPT distinguishers D and sufficiently large κ:

$$\left|\Pr\left[D(X, \langle P(X, W), V^*(X)\rangle) = 1\right] - \Pr\left[D(X, \mathsf{Sim}(X)) = 1\right]\right| \leq \mathsf{negl}(\kappa) \tag{4.1}$$

where the probabilities are taken over the outputs $(X, W) \leftarrow \mathsf{Gen}(1^\kappa)$, $(X, W) \in \mathcal{R}$.

Proof of Knowledge (PoK): There exists a polynomial-time extraction algorithm $\mathsf{Extract}$ that, given an arbitrary statement X and access to any prover P^*, outputs a witness W such that $(X, W) \in \mathcal{R}$ with probability at least $\Pr[\langle P^*, V\rangle = 1] - \varepsilon$ for knowledge error ε.

It is a known result (e.g. by Hazay and Lindell [HL10, Theorem 6.3.2]) that a correct and 2-special sound sigma protocol with challenge length t is a Proof of Knowledge with knowledge error 2^{-t}. In this chapter, this will generally equate to a sigma protocol using single-bit challenges, repeated with t iterations.

4.2 Previous SIDH identification scheme

Along with the SIDH key exchange protocol discussed in Section 1.3, De Feo, Jao, and Plût [DJP14] also introduced an isogeny-based identification scheme using a similar structure and ideas. We now recall that original scheme, and examine an issue with the proof of soundness provided by De Feo et al. for it. We then provide an explicit counterexample to the 2-special soundness proof of the scheme.

4.2.1 De Feo–Jao–Plût scheme

In an identical manner to SIDH, we let p be a large prime of the form $p = \ell_1^{e_1}\ell_2^{e_2}f \pm 1$, where ℓ_1 and ℓ_2 are distinct small primes. We then fix a supersingular elliptic curve E_0 defined over $\mathbb{F}_{p^2}$ with $\#E_0(\mathbb{F}_{p^2}) = (\ell_1^{e_1}\ell_2^{e_2}f)^2$. A private key is a random point $K_\phi \in E_0(\mathbb{F}_{p^2})$ of exact order $\ell_1^{e_1}$, or equivalently an isogeny $\phi : E_0 \to E_1$ of degree $\ell_1^{e_1}$ (where $E_1 = E_0/\langle K_\phi\rangle$).

In the identification scheme, we only make use of bases of order-$\ell_2^{e_2}$-torsion subgroups on each curve involved. Because of this, we use slightly different notation than in the description of the SIDH key exchange protocol: the subscripts of points will correspond to the elliptic curve they lie on. In this respect, we denote by P_0, Q_0 a fixed basis of the torsion subgroup $E_0[\ell_2^{e_2}] = \langle P_0, Q_0\rangle$.

The public parameters are, then, $\mathsf{pp} = (\ell_1, \ell_2, e_1, e_2, p, E_0, P_0, Q_0)$. Public keys have an identical form to SIDH keys. The De Feo–Jao–Plût identification scheme for public key $(E_1, \phi(P_0), \phi(Q_0))$ proceeds as follows:

1. **Commitment**: The prover chooses a random primitive $\ell_2^{e_2}$-torsion point K_ψ and writes

   $$K_\psi = [a]P_0 + [b]Q_0$$

   for some integers $0 \leq a, b < \ell_2^{e_2}$ not both divisible by ℓ_2. These same integers can be used to compute

   $$K_{\psi'} = \phi(K_\psi) = [a]\phi(P_0) + [b]\phi(Q_0).$$

   The prover uses Vélu's formulae to compute degree-$\ell_2^{e_2}$ isogenies ψ and ψ' whose kernels are generated by K_ψ and $K_{\psi'}$ respectively. Let the respective codomains of these isogenies be denoted by

   $$\begin{aligned} E_2 &= E_0/\langle K_\psi \rangle, \\ E_3 &= E_1/\langle \phi(K_\psi) \rangle = E_0/\langle K_\psi, K_\phi \rangle. \end{aligned}$$

   Finally, the prover computes $K_{\phi'} = \psi(K_\phi)$ and the corresponding isogeny $\phi' : E_2 \to E_3$. This gives the diagram in Figure 4.1.

   $$\begin{tikzcd} E_0 \arrow[r, dashed, "\phi"] \arrow[d, "\psi"'] & E_1 \arrow[d, "\psi'"] \\ E_2 \arrow[r, "\phi'"'] & E_3 \end{tikzcd}$$

   Figure 4.1: Commutative diagram of the SIDH identification scheme.

   The prover sends commitment $\mathsf{com} = (E_2, E_3)$ to the verifier.

2. **Challenge**: The verifier challenges the prover with a random bit $\mathsf{chall} \leftarrow \{0, 1\}$.

3. **Response**:

   - If $\mathsf{chall} = 0$, the prover reveals $\mathsf{resp} = (a, b)$, from which K_ψ and $\phi(K_\psi) = K_{\psi'}$ can be reconstructed.

   - If $\mathsf{chall} = 1$, the prover reveals $\mathsf{resp} = K_{\phi'}$.

In both cases, the verifier accepts the proof if the points revealed have the correct order and generate kernels of isogenies between the correct curves. We iterate this process t times to reduce the cheating probability to 2^{-t} (where t is chosen based on the security parameter κ).

Note that in an honest execution of the proof, we have

$$\widehat{\psi'} \circ \phi' \circ \psi = [\ell_2^{e_2}]\phi. \tag{4.2}$$

4.2.2 Issue with soundness proofs for the De Feo–Jao–Plût scheme

A vital aspect of security for an identification scheme, such as the De Feo–Jao–Plût scheme, is the soundness proof for the protocol. A proof of soundness for the De Feo–Jao–Plût scheme was given by multiple previous works [DJP14, YAJ+17, GPS20] based on the CSSI problem in Definition 2.2. A sketch of this soundness proof follows.

Suppose $\mathcal{A}$ is an adversary that takes as input the public key and succeeds in the identification protocol (all t iterations) with noticeable probability ϵ. Given a challenge instance (E, P, Q) for the CSSI problem with public parameters pp, we run $\mathcal{A}$ on the same tuple (E, P, Q) as the public key. In the first round, $\mathcal{A}$ outputs commitments $(E_{i,2}, E_{i,3})$ for $1 \leq i \leq t$. We then send a challenge $\mathsf{chall} \in \{0,1\}^t$ to $\mathcal{A}$ and, with probability ϵ, $\mathcal{A}$ outputs a response that satisfies the verification algorithm. Now, we use the standard replay technique: rewind $\mathcal{A}$ to the point just after it had output its commitments, but before it received the challenge, and then provide it with a different challenge $\mathsf{chall}' \in \{0,1\}^t$. With probability ϵ, $\mathcal{A}$ again outputs a valid response. Because the challenges differ in at least one bit, we must have a valid response for both $\mathsf{chall} = 0$ and 1 in at least one round of the sigma protocol (with the same commitment).

Let i be such an index where the challenges differ, so that $\mathsf{chall}_i \neq \mathsf{chall}'_i$. We now restrict our focus to this round and drop the subscript i. Thus, $\mathcal{A}$ has provided us with the commitment (E_2, E_3) and the responses $(\mathsf{resp}, \mathsf{resp}')$ for both challenges $\mathsf{chall} = 0$ and $\mathsf{chall} = 1$ successfully, as per the sigma protocol. Hence, $\mathcal{A}$ has provided the maps ψ, ϕ', ψ' in Figure 4.2.

$$\begin{tikzcd}
E_0 \arrow[r, dashed, "\phi"] \arrow[r, bend right, "\tilde{\phi}"'] \arrow[d, "\psi"'] & E_1 \arrow[d, "\psi'"] \\
E_2 \arrow[r, "\phi'"] & E_3
\end{tikzcd}$$

Figure 4.2: Information provided by adversary $\mathcal{A}$ after rewinding.

The extraction argument for 2-special soundness proceeds as follows: we have an explicit description of an isogeny $\tilde{\phi} = \widehat{\psi'} \circ \phi' \circ \psi$ from E_0 to E_1. The degree of $\tilde{\phi}$ is $\ell_1^{e_1}\ell_2^{2e_2}$. One can determine $\ker(\tilde{\phi}) \cap E_0[\ell_1^{e_1}]$ by iteratively testing points in $E_0[\ell_1^j]$ for $j = 1, 2, \ldots, e_1$. Hence, one determines the kernel of ϕ, as desired.

However, the important issue with this argument, which has so far gone unnoticed, is that it assumes $\ker(\phi) = \ker(\tilde{\phi}) \cap E_0[\ell_1^{e_1}]$. This assumption has no basis, and we will provide a simple counterexample to this argument in the following section. While we always recover *an* isogeny, it may not be ϕ at all—it is entirely possible the isogeny we recover does not even have codomain E_1. As a result, this proof of 2-special soundness is not valid.

4.2.3 Counterexample to soundness

Suppose we have fixed public parameters $\mathsf{pp} = (\ell_1, \ell_2, e_1, e_2, p, E_0, P_0, Q_0)$. Begin by generating a random $\ell_2^{e_2}$-torsion point $K_\psi \in E_0(\mathbb{F}_{p^2})$, and write $K_\psi = [a]P_2 + [b]Q_2$ for some integers $0 \leq a, b < \ell_2^{e_2}$ not both divisible by ℓ_2. Let $\psi : E_0 \to E_2$ be an isogeny with kernel generated by K_ψ.

Next, choose a random point $K_{\phi'} \in E_2$ of order $\ell_1^{e_1}$, and let $\phi' : E_2 \to E_3$ be a degree-$\ell_1^{e_1}$ isogeny with kernel generated by $K_{\phi'}$. In the same manner, choose a third random isogeny $\psi' : E_3 \to E_1$ of degree $\ell_2^{e_2}$. Then choose points $P_1, Q_1 \in E_1(\mathbb{F}_{p^2})$ such that $\ker(\widehat{\psi'}) = \langle [a]P_1 + [b]Q_1 \rangle$. Finally, publish (E_1, P_1, Q_1) as a public key.

In other words, we have:

$$E_0 \xrightarrow{\psi} E_2 \xrightarrow{\phi'} E_3 \xrightarrow{\psi'} E_1.$$

Now there is no reason to believe that an isogeny from E_0 to E_1 of degree $\ell_1^{e_1}$ exists, yet we can successfully respond to both challenge bits 0 and 1 in a single round of the identification scheme. This violates the 2-special soundness of the sigma protocol. Pulling back the kernel of ϕ' via ψ to E_0 will result in the kernel of an isogeny which, in general, will not have codomain E_1 (but instead another random curve). This is because ψ' is entirely unrelated to ψ in this case (they are not "parallel"), so we have no SIDH square.

A key observation is that a verifier could be fooled into accepting this public key by a prover who always uses the same curves (E_2, E_3) instead of randomly chosen ones. When $\mathsf{chall} = 0$ the prover responds with the pair (a, b) corresponding to the kernel of ψ and $\widehat{\psi}'$, and when $\mathsf{chall} = 1$ the prover responds with $K_{\phi'}$. The verifier will agree that all responses are correct and will accept the proof.

It is true that the verifier could test whether the commitments (E_2, E_3) are being reused, but this has never been stated as a requirement in any of the protocol descriptions. To tweak the verification protocol we need to know how "random" the pairs (E_2, E_3) (or, more realistically, the pairs (a, b)) need to be. One may think that the original scheme seems to be secure despite the issue with the proof, as long as the commitment (E_2, E_3) is not reused every time. However, in experiments with small primes, it is entirely possible to construct instances where even with multiple different commitments, a secret isogeny of the correct degree between E_0 and E_1 does not exist. We expect that this extrapolates to large primes too, although one could potentially argue that finding enough such instances is computationally infeasible.

It is also true that repeating (E_2, E_3) means the protocol is no longer zero-knowledge. We emphasise that soundness and zero-knowledge are independent security properties, which are proved separately (and affect different parties: one gives an assurance to the verifier and the other to the prover). The counterexample we have provided is a counterexample to the soundness proof. The fact that the counterexample is not consistent with the proof that the protocol is zero-knowledge is irrelevant.

Finally, one could consider basing security of the protocol on the general isogeny problem (Definition 2.1) because, even in our counterexample, an isogeny $E_0 \to E_1$ exists and can be extracted—it just does not have degree $\ell_1^{e_1}$. We find it interesting that none of the previous authors chose to do it that way. However, some applications may require using the identification or signature protocols to prove that an SIDH public key is well-formed, implying the secret isogeny has the correct degree. For such applications, we need soundness to be rigorously proved rather than the relation to be weakened.

The issue in the security proofs in the literature is not only that it is implicitly assumed that there is an isogeny of degree $\ell_1^{e_1}$ between E_0 and E_1. The key issue is that it is implicitly assumed that the pullback under ψ of $\ker(\phi')$ is the kernel of this isogeny. Our counterexample calls these assumptions into question and shows that the proofs are incorrect as written.

To make this very clear, consider again the soundness proof from De Feo et al. [DJP14]. The following diagram is provided within their proof. It implicitly assumes that the horizontal isogeny ϕ' has kernel given by $\psi(S)$, so that the image curve is $E/\langle S, R\rangle$.

$$\begin{tikzcd}
E \arrow[d, "\psi"'] & E/\langle S\rangle \arrow[d, "\psi'"] \\
E/\langle R\rangle \arrow[r, "\phi'"'] & E/\langle S, R\rangle
\end{tikzcd}$$

This implicit assumption seems to have been repeated in all subsequent works, such as by Yoo et al. [YAJ$^+$17] and Galbraith et al. [GPS20].

4.3 New SIDH zero-knowledge proof scheme

Let $\mathsf{pp} = (p, \ell_1, \ell_2, e_1, e_2, E_0, P_0, Q_0)$ be public parameters such that $E_0(\mathbb{F}_{p^2})[\ell_2^{e_2}] = \langle P_0, Q_0\rangle$. As before, suppose a user has a secret isogeny $\phi : E_0 \to E_1$ of degree $\ell_1^{e_1}$ with kernel $\ker(\phi) = \langle K_\phi\rangle$.

We propose a new sigma protocol to prove knowledge of this isogeny given the public key $(E_1, P_1 = \phi(P_0), Q_1 = \phi(Q_0))$. The protocol is presented in Figure 4.4. $\mathsf{IsogenyFromKernel}$ is a function taking a kernel point and outputting an isogeny and codomain curve with said kernel. $\mathsf{CanonicalBasis}_2$ is a deterministic function taking a curve and outputting an $\ell_2^{e_2}$-torsion basis on the given curve. $\mathsf{DualKernel}$ is a function taking an isogeny ψ and outputting a generator $K_{\widehat{\psi}}$ of the dual isogeny $\widehat{\psi}$ (c.f. Lemma 1.7). Figure 4.3 shows the commutative diagram of the sigma protocol.

Intuitively, the scheme follows Section 4.2.1, with a single bit challenge—if the challenge is 0, we reveal the vertical isogenies ψ and ψ', while if the challenge is 1, we reveal the horizontal ϕ'. The difference is the introduction of additional points on E_3 to the commitment, which force ψ, ψ' to be, in some sense, "compatible" or "parallel". This restriction allows the proof of 2-special soundness to work.

We then repeat the sigma protocol t times in parallel (where t is chosen based on the security parameter κ) and set com to be the concatenation of all individual $[\mathsf{com}_i]_{i\in\{1,\ldots,t\}}$ for each iteration i, $\mathsf{chall} = [\mathsf{chall}_i]_{i\in\{1,\ldots,t\}}$ and $\mathsf{resp} = [\mathsf{resp}_i]_{i\in\{1,\ldots,t\}}$.

$$\begin{tikzcd}
E_0 \arrow[r, dashed, "\phi"] \arrow[d, "\psi"'] & E_1 \arrow[d, "\psi'"] \\
E_2 \arrow[r, "\phi'"'] & E_3
\end{tikzcd}$$

Figure 4.3: Commutative diagram of the SIDH identification scheme, as in Figure 4.1. Here, points $P_3, Q_3 = \phi'(P_2), \phi'(Q_2)$ are provided on E_3, where $P_2, Q_2 \leftarrow \mathsf{CanonicalBasis}(E_2)$.

Remark 4.3. There are certainly improvements that can be made to increase efficiency and compress the size of signatures, but many of these are standard, and we will not explore them

Round 1 (commitment)

```
Sample random $\ell_2^{e_2}$-isogeny kernel $\langle K_\psi \rangle \subset E_0$
Write $K_\psi = [a]P_0 + [b]Q_0 \in E_0$ for $a, b \in \mathbb{Z}/\ell_2^{e_2}\mathbb{Z}$
$K_{\psi'} := \phi(K_\psi) = [a]\phi(P_0) + [b]\phi(Q_0) \in E_1$
$\psi, E_2 \leftarrow \mathsf{IsogenyFromKernel}(K_\psi)$
$P_2, Q_2 \leftarrow \mathsf{CanonicalBasis}_2(E_2)$
$K_{\phi'} := \psi(K_\phi) \in E_2$
$\phi', E_3 \leftarrow \mathsf{IsogenyFromKernel}(K_{\phi'})$
$P_3, Q_3 := \phi'(P_2), \phi'(Q_2) \in E_3$
Prover sends $\mathsf{com} := (E_2, E_3, P_3, Q_3)$ to the verifier
```

Round 2 (challenge)

```
Verifier sends $\mathsf{chall} \leftarrow \{0, 1\}$ to the prover
```

Round 3 (response)

```
if $\mathsf{chall} = 1$ then
    $\mathsf{resp} := K_{\phi'}$
else
    $K_{\widehat{\psi}} \leftarrow \mathsf{DualKernel}(\psi)$
    Write $K_{\widehat{\psi}} = [c]P_2 + [d]Q_2$ for $c, d \in \mathbb{Z}/\ell_2^{e_2}\mathbb{Z}$
    $\mathsf{resp} := (c, d)$
Prover sends $\mathsf{resp}$ to the verifier
```

Verification

```
$(E_2, E_3, P_3, Q_3) \leftarrow \mathsf{com}$
if $\mathsf{chall} = 1$ then
    $K_{\phi'} \leftarrow \mathsf{resp}$
    Check $K_{\phi'}$ has order $\ell_1^{e_1}$ and lies on $E_2$, otherwise output $\mathsf{reject}$
    $P_2, Q_2 \leftarrow \mathsf{CanonicalBasis}_2(E_2)$
    $\phi', E_3' \leftarrow \mathsf{IsogenyFromKernel}(K_{\phi'})$
    Verify $E_3 = E_3'$ and $P_3, Q_3 = \phi'(P_2), \phi'(Q_2)$, otherwise output $\mathsf{reject}$
else
    $(c, d) \leftarrow \mathsf{resp}$
    $P_2, Q_2 \leftarrow \mathsf{CanonicalBasis}_2(E_2)$
    $K_{\widehat{\psi}} := [c]P_2 + [d]Q_2$
    $K_{\widehat{\psi}'} := [c]P_3 + [d]Q_3$
    Check $K_{\widehat{\psi}}$, $K_{\widehat{\psi}'}$ have order $\ell_2^{e_2}$, otherwise output $\mathsf{reject}$
    $\widehat{\psi}, E_0' \leftarrow \mathsf{IsogenyFromKernel}(K_{\widehat{\psi}})$
    $\widehat{\psi}', E_1' \leftarrow \mathsf{IsogenyFromKernel}(K_{\widehat{\psi}'})$
    Check $E_0 = E_0'$ and $E_1 = E_1'$, otherwise output $\mathsf{reject}$
Output $\mathsf{accept}$
```

Figure 4.4: One iteration of the sigma protocol for our new SIDH zero-knowledge proof and identification scheme. The public parameters are $\mathsf{pp} = (p, \ell_1, \ell_2, e_1, e_2, E_0, P_0, Q_0)$. The public key is (E_1, P_1, Q_1), and the corresponding secret isogeny is ϕ.

here. For example, in practice, the commitment information (E_3, P_3, Q_3) would be replaced with a triplet of three x-coordinates of points, as in SIKE [ACC+17]. Alternatively, P_3, Q_3 could be represented in terms of $\mathsf{CanonicalBasis}_2(E_3)$ using a compressed change-of-basis matrix.

Theorem 4.4. *The sigma protocol in Figure 4.4 for relation*

$$\mathcal{R}_{\mathsf{weakSIDH}} = \left\{((E_1, P_1, Q_1), \phi) \mid \phi : E_0 \to E_1,\ \deg(\phi) = \ell_1^{e_1}\right\} \tag{4.3}$$

is correct, 2-special sound, and distributionally zero-knowledge (with respect to the output distribution of the uniform SIDH key generation algorithm) assuming the DMSIDH and DSSP problems are hard. Repeated with κ iterations, it is thus a Proof of Knowledge for $\mathcal{R}_{\mathsf{weakSIDH}}$ with knowledge error $2^{-\kappa}$.

Proof. We prove the three properties of Theorem 4.4 separately below.

Correctness: Following the protocol honestly will result in an accepting transcript. This is clear for the $\mathsf{chall} = 1$ case. For the $\mathsf{chall} = 0$ case, observe that

$$\begin{aligned} \phi'(K_{\widehat{\psi}}) &= \phi'([c]P_2 + [d]Q_2) \\ &= [c]P_3 + [d]Q_3 \\ &= K_{\widehat{\psi}'}, \end{aligned} \tag{4.4}$$

so $K_{\widehat{\psi}'}$ generates the kernel of $\widehat{\psi}'$.

2-special soundness: Without loss of generality, suppose we obtain two sigma protocol transcripts $(\mathsf{com}, 0, \mathsf{resp})$, $(\mathsf{com}, 1, \mathsf{resp}')$, for example, by rewinding an adversary $\mathcal{A}$ after it outputs com and providing a different challenge. Then recover $(c, d) \leftarrow \mathsf{resp}$ and $K_{\phi'} \leftarrow \mathsf{resp}'$, and let ϕ' be an isogeny whose kernel is generated by $K_{\phi'}$. Applying Lemma 1.8, with $(\phi_A, \phi_B, \phi_{AB}) := (\phi', \widehat{\psi}, \widehat{\psi}')$, we obtain an isogeny $\chi : E_0 \to E_1$ of degree $\ell_1^{e_1}$. The conditions of the lemma on the kernels of $\widehat{\psi}$ and $\widehat{\psi}'$ are satisfied because $\phi'(K_{\widehat{\psi}}) = K_{\widehat{\psi}'}$, as above.

This shows the protocol is 2-special sound, and that it is a Proof of Knowledge of an isogeny corresponding to the given public key curve. Because this protocol does not guarantee correctness of the points P_1, Q_1 in the public key (as briefly discussed at the end of this section), this is only a proof for the $\mathsf{weakSIDH}$ relation. In the next section, we will modify this protocol further to also include these torsion points in the relation.

Zero-knowledge: Proof of distributional zero-knowledge (with respect to the usual SIDH key generation algorithm Gen) follows as in [DJP14]. Let V^* be a cheating verifier, which shall be used as a black box by the simulator Sim. Let $((E_1, P_1, Q_1), \phi) \leftarrow \mathsf{Gen}$ be an SIDH key pair. We show that Sim can generate a valid transcript for t iterations of the protocol with respect to statement (E_1, P_1, Q_1). At each step, Sim makes a guess as to what the next challenge bit chall will be, and then proceeds as follows:

- If $\mathsf{chall} = 0$, Sim simulates as per the honest protocol by choosing a random kernel $\langle K_\psi \rangle$ on E_0 of order $\ell_2^{e_2}$, writing $K_\psi = [a]P_0 + [b]Q_0$ for $a, b \in \mathbb{Z}/\ell_2^{e_2}\mathbb{Z}$, and setting $K_{\psi'} := [a]P_1 + [b]Q_1$ on E_1. Sim computes the two vertical isogenies $\psi : E_0 \to E_2, \psi' : E_1 \to E_3$ from these kernel generators respectively. The simulator then computes the corresponding dual isogenies and the canonical basis $P_2, Q_2 \leftarrow \mathsf{CanonicalBasis}_2(E_2)$. Let $K_{\widehat{\psi}}$ and $K_{\widehat{\psi}'}$ be generators of the kernels of $\widehat{\psi}$ and $\widehat{\psi}'$ respectively. The simulator writes $K_{\widehat{\psi}}$ in terms of the canonically-generated basis on E_2 as $[c]P_2 + [d]Q_2$, then chooses a torsion basis on E_3 as $P_3, Q_3 \in E_3$ in such a way that these points P_3, Q_3 are indistinguishable from points chosen in an honest protocol transcript:

1. Obtain a point $S \in E_2$ and its image $S' = \phi'(S_2)$ via the algorithm in Figure 1.1 despite ϕ' being unknown (c.f. Lemma 1.8).
2. Choose any $T \in E_2$ of order $\ell_2^{e_2}$ such that $E_2[\ell_2^{e_2}] = \langle S, T\rangle$.
3. Choose a point $T' \in E_3$ such that $E_3[\ell_2^{e_2}] = \langle S', T'\rangle$, and such that $e_{\ell_2^{e_2}}(S,T)^{\ell_1^{e_1}} = e_{\ell_2^{e_2}}(S', T')$.
4. Solve discrete logarithms of P_2, Q_2 with respect to S, T on E_2 to obtain a change-of-basis matrix, and apply the same change of basis to S', T' on E_3 to obtain points P_3, Q_3.

Note that the above operations are efficient due to the ease of computing discrete logarithms when the group order is very smooth [Tes99].

- If $\mathsf{chall} = 1$, the simulator chooses a random supersingular elliptic curve[1] E_2 and a random point $K_{\phi'} \in E_2$ of order $\ell_1^{e_1}$. Sim then computes an isogeny $\phi' : E_2 \to E_3$ with kernel $\langle K_{\phi'}\rangle$. Finally, the simulator generates a canonical basis $P_2, Q_2 \leftarrow \mathsf{CanonicalBasis}_2(E_2)$, computes $P_3, Q_3 := \phi'(P_2), \phi'(Q_2)$, and sets the commitment to (E_2, E_3, P_3, Q_3) and the response to $K_{\phi'}$.

After providing com to V^*, if the challenge bit that V^* outputs is not the same as Sim's guess, Sim simply discards that iteration and runs again. Sim stops whenever V^* rejects or after t successful rounds. Suppose the probability of V^* not choosing the same bit as Sim's guess is noticeably different from 1/2. Then V^* can be used as a distinguisher for the DSSP problem (in fact, an even harder problem than the DSSP where, instead of the isogeny ϕ', only its action on $E_2[\ell_2^{e_2}]$ is given). We show this below, in the $\mathsf{chall} = 1$ case of this proof. So the probability Sim guesses correctly each round is exponentially close to 1/2 if the DSSP problem is hard. Thus, Sim will run in polynomial-time.

To prove the indistinguishability of simulated transcripts from true interactions of a prover P with V^*, it is enough to show that one round of the sigma protocol is indistinguishable (by the hybrid technique of Goldreich, Micali, and Wigderson [GMW91]).

When $\mathsf{chall} = 0$, the choice of ψ and ψ' is done exactly as in the honest protocol, so the curves E_2, E_3 in the commitment are perfectly indistinguishable from those in honest transcripts. We show that the points P_3, Q_3 are also indistinguishable, assuming the DMSIDH problem is hard. Suppose $\mathcal{B}_0$ is a PPT adversary which can distinguish between the simulation and the real transcripts for $\mathsf{chall} = 0$ with advantage Adv_0. Let $((E_0, P_0, Q_0), (E_1, \phi(P_0), \phi(Q_0)), \psi, \psi', S, T, \phi'(S), T')$ be a challenge instance of the DMSIDH problem. Denote by E_2 the codomain of ψ, and E_3 the codomain of ψ'. Set $P_2, Q_2 \leftarrow \mathsf{CanonicalBasis}_2(E_2)$, and proceed as in Step 4 of the simulation above to obtain points P_3, Q_3 from S', T' using a change of basis matrix $A = (a_i)$ derived from (P_2, Q_2) and (S, T):

$$\begin{pmatrix} P_2 \\ Q_2 \end{pmatrix} = \begin{pmatrix} a_0 & a_1 \\ a_2 & a_3 \end{pmatrix} \cdot \begin{pmatrix} S \\ T \end{pmatrix} \tag{4.5}$$

[1] One way to do so is to take a random ℓ_2-isogeny walk from E_0. To ensure a distribution close to uniform, we take a walk of length $\gtrsim \log(p) \approx 2e_2$. However, a walk of length e_2 is sufficient to get a variant of DSSP that is also believed to be hard.

Write the kernel of $\widehat{\psi}$ as $\ker(\widehat{\psi}) = [c]P_2 + [d]Q_2$ for scalars c, d. Finally, give to $\mathcal{B}_0$ the transcript

$$\begin{aligned} \mathsf{com} &:= (E_2, E_3, P_3, Q_3), \\ \mathsf{chall} &:= 0, \\ \mathsf{resp} &:= (c, d). \end{aligned} \tag{4.6}$$

If $T' = \phi'(T)$ in the challenge instance of the DMSIDH problem (i.e., if the instance is from distribution $\mathcal{D}_0$), then we necessarily have that $P_3 = [a_0]\phi'(S) + [a_1]\phi'(T) = \phi'(P_2)$, and similarly $Q_3 = \phi'(Q_2)$. Hence, the distribution of transcripts will be identical to the honest protocol. On the other hand, the transcript simulator selects a random T' such that $E_3[\ell_2^{e_2}] = \langle S', T' \rangle$ and $e_{\ell_2^{e_2}}(S, T)^{\ell_1^{e_1}} = e_{\ell_2^{e_2}}(S', T')$. Let $T' = [q]\phi'(T) + [r]\phi'(S) = [q]\phi'(T) + [r]S'$. The pairing condition gives $e_{\ell_2^{e_2}}(S', [q]\phi'(T) + [r]S') = e_{\ell_2^{e_2}}(S', \phi'(T))^q$ implying $q = 1$. Hence, $T' = \phi'(T + [r]S)$. Then, because the transcript simulator behaves identically to the reduction in computing P_3, Q_3 (via applying the same change of basis matrix to S', T'), the transcript distribution in the reduction will be identical to the transcripts generated by the simulator. Therefore, the response from $\mathcal{B}_0$ will solve the DMSIDH problem with advantage Adv_0.

When $\mathsf{chall} = 1$, we consider the distribution of (E_2, E_3, ϕ'). While this distribution is not correct a priori, the DSSP computational assumption in Definition 2.5 implies it is computationally hard to distinguish the simulation from the real game (as in the proof in [GPS20]). Because the action of ϕ' on canonical basis $P_2, Q_2 \in E_2$ can be computed by any party who knows ϕ', the distribution of (E_2, E_3, P_3, Q_3) must also be indistinguishable between simulation and real transcripts.

Suppose $\mathcal{B}_1$ is a PPT adversary which can distinguish between the simulation and the real transcripts for $\mathsf{chall} = 1$ with advantage Adv_1. Given an instance of the DSSP problem, (E_2, E_3, ϕ'), compute $P_2, Q_2 \leftarrow \mathsf{CanonicalBasis}_2(E_2)$. Then let $P_3 := \phi'(P_2)$ and $Q_3 := \phi'(Q_2)$, and set

$$\begin{aligned} \mathsf{com} &:= (E_2, E_3, P_3, Q_3), \\ \mathsf{chall} &:= 1, \\ \mathsf{resp} &:= K_{\phi'}, \end{aligned} \tag{4.7}$$

where $K_{\phi'}$ is any generator of $\ker(\phi')$.

$\mathcal{B}_1$, given $(\mathsf{com}, \mathsf{chall}, \mathsf{resp})$, will then solve the DSSP with the same advantage Adv_1. It is for this same reason that a cheating verifier V^* is unable to distinguish based on com alone whether the simulator is attempting a $\mathsf{chall} = 0$ or $\mathsf{chall} = 1$ simulation with non-negligible advantage, if it cannot solve the DSSP problem with non-negligible advantage.

Hence, the scheme is computationally zero-knowledge assuming the DSSP and DMSIDH problems are hard. □

Remark 4.5. We note that the points P_1, Q_1 are not actually used in the verification algorithm, so could be omitted entirely in practice if desired. After observing just two honest iterations of the sigma protocol, on average, the verifier would be able to reconstruct (P_1, Q_1). This idea is made more rigorous in the Section 4.4, where instead of recovering the points, we prove their correctness instead. There, too, the points could actually be omitted from the public key since their correct recovery is guaranteed. We cannot remove the points entirely from the protocol in theory, though, because they are required for the simulator in the proof of zero-knowledge. Because the proof of zero-knowledge assumes indistinguishability from a prover who behaves honestly, the points used by the simulator are required to be correct.

Remark 4.6. If there was an efficient solution to the computational version of the DMSIDH problem—that is, the problem of finding the correct image of T under the secret ϕ'—then we could obviously simulate perfectly in the proof of zero-knowledge. Moreover, if there did exist an efficient distinguisher for the DMSIDH problem, then integrating it into the verification step of the protocol in Figure 4.4 would be enough to prove the strong relation that we will define in Section 4.4.

It would be surprising if there was a gap between DMSIDH and its computational analogue, leading to an efficient, but not zero-knowledge, protocol for both the weak and the strong relation. Our intuition tells us that such a gap should not exist, but we have been unable to prove it.

Why this protocol does not prove the correctness of the points (P_1, Q_1)

We briefly explain why the protocol in this section does not convince a verifier that $(P_1, Q_1) = (\phi(P_0), \phi(Q_0))$. The first observation is that Figure 4.4 does not actually use P_1 or Q_1 anywhere, so of course, nothing is proved. But one could tweak the protocol in the $\mathsf{chall} = 0$ case to use the isogenies $\widehat{\psi} : E_2 \to E_0$ and $\widehat{\psi}' : E_3 \to E_1$ to test the points. For example, computing the duals of these isogenies one could compute integers (a, b) such that $\ker(\psi) = \langle [a]P_0 + [b]Q_0 \rangle$ and then test whether $\ker(\psi') = \langle [a]P_1 + [b]Q_1 \rangle$.

The problem for the verifier is that this is not enough to deduce that $(P_1, Q_1) = (\phi(P_0), \phi(Q_0))$. For example, a dishonest prover who wants to perform an attack might set $(P_1, Q_1) = (\phi(P_0), \phi(Q_0) + T)$ where T is a point of order ℓ_2. If the prover always uses integers b that are multiples of ℓ_2 then this cheating will not be detected by the verifier. Hence, the protocol needs to be changed so that the verifier can tell that the kernels of the isogenies $\widehat{\psi}$ are sufficiently independent across the executions of the protocol. This is the fundamental problem that we solve in the next section.

4.4 Correctness of the points in an SIDH public key

We have shown in Section 4.3 that successful completion of the new sigma protocol indeed proves knowledge of a degree-$\ell_1^{e_1}$ isogeny from E_0 to E_1 (as per the relation $\mathcal{R}_{\mathsf{weakSIDH}}$ in Theorem 4.4). However, an SIDH public key (E_1, P_1, Q_1) also consists of the two torsion points, and these points are the cause of issues such as the adaptive attack [GPST16], as discussed in Section 2.1. In this section, we show that the choice of points P_1, Q_1 by a malicious prover is severely restricted if they must keep them consistent with "random enough" values of a, b (i.e., random choices of ψ)—preventing adaptive attacks entirely. This gives the following stronger SIDH relation:

$$\mathcal{R}_{\mathsf{SIDH}} = \left\{ ((E_1, P_1, Q_1),\, \phi) \;\middle|\; \begin{array}{c} \phi : E_0 \to E_1,\ \deg(\phi) = \ell_1^{e_1}, \\ P_1 = \phi(P_0),\ Q_1 = \phi(Q_0) \end{array} \right\}. \tag{4.8}$$

Figures 4.5 and 4.6 show the modified protocol proving knowledge of a witness in this strong relation.

Let us reconsider the protocol in Figure 4.4 for a moment. We have that $\ker(\widehat{\psi}') = \phi'(\ker(\widehat{\psi}))$ by the 2-special soundness of Theorem 4.4. Applying the algorithm in Figure 1.1 to $(E_2, P_2, Q_2, E_3, P_3, Q_3, E_0, E_1, \widehat{\psi}, \widehat{\psi}')$ gives us a pair $(R_0, R_1 = \chi(R_0))$ for $\chi : E_0 \to E_1$, where $\ker(\chi) = \widehat{\psi}(\ker(\phi'))$. Note that ϕ in the algorithm and Lemma 1.8 corresponds to ϕ' here because we have flipped the SIDH square "upside down." Because the degrees of ϕ' and $\widehat{\psi}$ are coprime, we can translate this to $\psi(\ker(\chi)) = \ker(\phi')$. Note that R_0 and R_1 will be scalar multiples (by the same scalar) of the K_ψ and $K_{\psi'}$ used by the prover in the commitment round of the protocol.

Consequently, two (honest) answers to $\mathsf{chall} = 0$ reveal two pairs of points $R_{1,0}, R_{1,1} = \chi(R_{0,0}), \chi(R_{0,1})$. If these are independent, they fix the action of χ on the whole $\ell_2^{e_2}$-torsion (as a basis for the $\ell_2^{e_2}$-torsion subgroups on both curves), and then if this action is the same as the action of ϕ, we must have that $\chi = \phi$. The easiest way to enforce two such honest answers is to "double" the protocol. Thus, in each round of our new sigma protocol, we shall commit to two SIDH squares rather than just one, and require that the kernel generators of the ψ's in these two squares are independent of each other. We add this independence as an extra check during verification. We also require an assurance that both squares use consistent isogenies ϕ'. For this purpose we use a random $\ell_1^{e_1}$-torsion basis (U, V) on E_0 and compute the image of this basis on both curves $E_{2,i}$—if both ϕ'_0 and ϕ'_1 are the images of ϕ under the vertical isogenies ψ_i, then both should be representable in terms of $(\psi_i(U), \psi_i(V))$ using the same coefficients. These extra checks achieve a 2-special sound protocol for the stronger SIDH relation above. We stress that the points U, V are not made public in the commitment. In the following protocol, $\mathsf{RandomBasis}_1$ is a function taking a curve and outputting a random pair of points U, V which generate the $\ell_1^{e_1}$-torsion subgroup on the given curve. The function $\mathsf{RandomBasis}_1$ is called many times on the same curve E_0 during t rounds of the protocol, and it is important that the outputs are independent and not known to the verifier in the $\mathsf{chall} = 1$ case.

Theorem 4.7. *For a fixed security parameter κ and SIDH public key (E, P, Q), a proof consisting of κ iterations of the sigma protocol in Figure 4.5 is a distributionally zero-knowledge (with respect to the usual SIDH key generation algorithm) Proof of Knowledge for $\mathcal{R}_{\mathsf{SIDH}}$ with knowledge error $2^{-\kappa}$, assuming the DMSIDH and Double-DSSP problems are hard.*

Proof. Again we prove correctness, soundness, and zero-knowledge individually.

Correctness: As mentioned above, the point $R_{0,i}$ will always be an invertible scalar multiple of the point K_ψ used by the prover in the commitment round (in the i-th SIDH square) of the protocol, because both K_ψ and $R_{0,i}$ are generators of the kernel of ψ_i in the i-th SIDH square. This implies the pair (a'_i, b'_i) is an invertible scalar multiple of (a_i, b_i). Hence, because the honest prover will use commitments such that $a_0b_1 - a_1b_0$ is invertible, then a'_i, b'_i necessarily exist such that $a'_0b'_1 - a'_1b'_0$ is invertible in line 22 of verification. Also note that because $K_{\phi',i} = [e]U'_i + [f]V'_i = [e]\psi_i(U) + [f]\psi_i(V)$ for both $i \in \{0, 1\}$, and U, V have order coprime to the degree of ψ_i, the checks involving U'_i, V'_i, e, and f will also succeed. The correctness of the rest of the protocol can also be verified in a straightforward way.

Zero-knowledge: Let V^* be a cheating verifier, and $((E_1, P_1, Q_1), \phi) \leftarrow \mathsf{Gen}$ a randomly sampled SIDH key pair. Sim will generate a valid transcript for t iterations of the protocol as follows: at each step, Sim will make a guess as to what the next challenge bit chall will be, and proceeds appropriately:

- If $\mathsf{chall} = 0$, Sim will behave as in the proof of Theorem 4.4 to generate the first SIDH square arbitrarily. The simulator will then generate a second SIDH square in almost the same way, but ensuring that the second ψ chosen uses kernel coefficients independent of those used in the first square (just like the honest prover would do in the commitment round of Figure 4.5). Sim will also randomly generate a basis (U, V) of the $\ell_1^{e_1}$-torsion on E_0 and compute the images $U'_i, V'_i = \psi_i(U), \psi_i(V)$ exactly as in Figure 4.5. The commitment and response will be formed exactly as in the honest protocol.

Round 1 (commitment)

```
Run **commitment** from Figure 4.4, giving commitment $\mathsf{com}_0 = (E_{2,0}, E_{3,0}, P_{3,0}, Q_{3,0})$. Let
  $a_0, b_0$ be the coefficients used in Line 2 and $\psi_0$ be the isogeny from Line 4 (of Figure 4.4) of
  this execution.
Run **commitment** from Figure 4.4 again, subject to one extra condition:
    If $a_1, b_1$ are the coefficients used in Line 2 (of Figure 4.4) of this execution, then require
    $a_0 b_1 - a_1 b_0$ invertible modulo $\ell_2^{e_2}$. Otherwise, repeat Line 1 (of Figure 4.4).
  Let $\mathsf{com}_1 = (E_{2,1}, E_{3,1}, P_{3,1}, Q_{3,1})$ be the commitment returned by this execution, and $\psi_1$ be
  the isogeny from Line 4.
$U, V \leftarrow \mathsf{RandomBasis}_1(E_0)$
for $i \in \{0, 1\}$ do
    $U'_i := \psi_i(U)$
    $V'_i := \psi_i(V)$
Output $\mathsf{com} := (\mathsf{com}_0, U'_0, V'_0, \mathsf{com}_1, U'_1, V'_1)$
```

Round 2 (challenge)

- same as in Figure 4.4, outputting chall -

Round 3 (response)

```
if $\mathsf{chall} = 1$ then
    Write $K_\phi = [e]U + [f]V$ for $e, f \in \mathbb{Z}/\ell_1^{e_1}\mathbb{Z}$
    Output $\mathsf{resp} := (e, f)$
else
    for $i \in \{0, 1\}$ do
        $K_{\widehat{\psi}} \leftarrow \mathsf{DualKernel}(\psi)$
        Write $K_{\widehat{\psi}} = [c_i]P_2 + [d_i]Q_2$ for $c_i, d_i \in \mathbb{Z}/\ell_2^{e_2}\mathbb{Z}$
        $\mathsf{resp}_i := (c_i, d_i)$
    Output $\mathsf{resp} := (\mathsf{resp}_0, \mathsf{resp}_1)$
```

Figure 4.5: Modification of the Sigma protocol in Figure 4.4 to prove the stronger relation $\mathcal{R}_{\mathsf{SIDH}}$. Lines in grey are unchanged from Figure 4.4 to highlight the differences. The verification algorithm is given in Figure 4.6.

Verification

```
$(\mathsf{com}_0, U_0', V_0', \mathsf{com}_1, U_1', V_1') \leftarrow \mathsf{com}$
if $\mathsf{chall} = 1$ then
    $(e, f) \leftarrow \mathsf{resp}$
    for $i \in \{0, 1\}$ do
        Recover $K_{\phi',i} := [e]U_i' + [f]V_i'$
        Verify $(\mathsf{com}_i, \mathsf{chall}, K_{\phi',i})$ as in Figure 4.4 verification
        If verification fails, output reject
else
    for $i \in \{0, 1\}$ do
        $(E_2, E_3, P_3, Q_3) \leftarrow \mathsf{com}_i$
        $(c, d) \leftarrow \mathsf{resp}_i$
        $P_2, Q_2 \leftarrow \mathsf{CanonicalBasis}_2(E_2)$
        $K_{\widehat{\psi}} := [c]P_2 + [d]Q_2$
        $K_{\widehat{\psi}'} := [c]P_3 + [d]Q_3$
        Check $K_{\widehat{\psi}}$, $K_{\widehat{\psi}'}$ have order $\ell_2^{e_2}$, otherwise output reject
        $\widehat{\psi}_i, E_0' \leftarrow \mathsf{IsogenyFromKernel}(K_{\widehat{\psi}})$
        $\widehat{\psi}_i', E_1' \leftarrow \mathsf{IsogenyFromKernel}(K_{\widehat{\psi}'})$
        Check $E_0 = E_0'$ and $E_1 = E_1'$, otherwise output reject
        Choose $(c', d')$ such that $c'd - d'c$ is invertible modulo $\ell_2^{e_2}$
        $R_{0,i} := \widehat{\psi}_i([c']P_2 + [d']Q_2)$
        $R_{1,i} := \widehat{\psi}_i'([c']P_3 + [d']Q_3)$
        Check there exist $a_i', b_i' \in \mathbb{Z}/\ell_2^{e_2}\mathbb{Z}$ such that, simultaneously,
          i. $R_{0,i} = [a_i']P_0 + [b_i']Q_0$,
          ii. $R_{1,i} = [a_i']P_1 + [b_i']Q_1$,
        otherwise output reject
    Check $\widehat{\psi}_0(U_0') = \widehat{\psi}_1(U_1')$ and $\widehat{\psi}_0(V_0') = \widehat{\psi}_1(V_1')$, otherwise output reject
    Check that $a_0'b_1' - a_1'b_0'$ is invertible modulo $\ell_2^{e_2}$, otherwise output reject
Output accept if all the above conditions hold
```

Figure 4.6: Modification of the verification algorithm in Figure 4.4 for the proof of the stronger relation $\mathcal{R}_{\mathsf{SIDH}}$. Lines in grey are unchanged from Figure 4.4 to highlight the differences.

- When $\mathsf{chall} = 1$, the behaviour of Sim is similar to the $\mathsf{chall} = 1$ simulation in the proof of Theorem 4.4, but repeated twice. First, Sim will choose two random curves $E_{2,i}$, for $i \in \{0, 1\}$. Sim will then choose a random point $K_{\phi',0} \in E_{2,0}$ of order $\ell_1^{e_1}$, and a random basis $\langle U_0', V_0' \rangle = E_{2,i}[\ell_1^{e_1}]$, and write $K_{\phi',0} = [e]U_0' + [f]V_0'$ for integers e, f. Next, Sim will randomly generate a basis (U_1', V_1') of the $\ell_1^{e_1}$-torsion subgroup on $E_{2,1}$ such that $e_{\ell_1^{e_1}}(U_0', V_0') = e_{\ell_1^{e_1}}(U_1', V_1')$, and let $K_{\phi',1} = [e]U_1' + [f]V_1'$. Let ϕ_0', ϕ_1' be isogenies with respective kernels $K_{\phi',0}, K_{\phi',1}$, and let $E_{3,i}$ be the codomain of ϕ_i'. Finally, the simulator generates canonical bases $P_{2,i}, Q_{2,i} \leftarrow \mathsf{CanonicalBasis}_2(E_{2,i})$, computes $P_{3,i}, Q_{3,i} := \phi_i'(P_{2,i}), \phi_i'(Q_{2,i})$, and sets $\mathsf{com} := \big((E_{2,i}, E_{3,i}, P_{3,i}, Q_{3,i}, U_i', V_i')_{i \in \{0,1\}}\big)$ and $\mathsf{resp} := (e, f)$.

After providing com to V^*, if the challenge bit that V^* outputs is not the same as Sim's guess, Sim simply discards that iteration and runs again. Sim stops whenever V^* rejects or after t successful rounds. Suppose the probability of V^* not choosing the same bit as Sim's guess is noticeably different from $1/2$. Then V^* can be used as a distinguisher for (a harder variant of) the Double-DSSP problem. This is analogous to the reasoning in the proof of Theorem 4.4, and follows directly from the proof of indistinguishability of $\mathsf{chall} = 1$ transcripts below. So the probability Sim guesses correctly each round is exponentially close to $1/2$ if the Double-DSSP problem is hard. Thus, Sim will run in polynomial-time.

Correctness of the simulator: We now show that the simulator will successfully generate valid transcripts with the additional R_0, R_1 check in the protocol. Suppose the verifier arbitrarily chooses c', d' such that $c'd - d'c$ is invertible modulo $\ell_2^{e_2}$, where c, d were used in the response of either square $i \in \{0, 1\}$. We have that

$$R_2 = \begin{pmatrix} c' & d' \end{pmatrix} \begin{pmatrix} P_2 \\ Q_2 \end{pmatrix} = \begin{pmatrix} c' & d' \end{pmatrix} A \begin{pmatrix} S \\ T \end{pmatrix}, \tag{4.9}$$

where the matrix A is the same change-of-basis matrix from Equation 4.5 of the proof of Theorem 4.4. So,

$$R_0 = \widehat{\psi}(R_2) = \begin{pmatrix} c' & d' \end{pmatrix} A \begin{pmatrix} \widehat{\psi}(S) \\ \widehat{\psi}(T) \end{pmatrix} = \begin{pmatrix} c' & d' \end{pmatrix} A \begin{pmatrix} \mathcal{O}_{E_0} \\ \widehat{\psi}(T) \end{pmatrix} \tag{4.10}$$

because S is in the kernel of $\widehat{\psi}$. Similarly,

$$R_3 = \begin{pmatrix} c' & d' \end{pmatrix} \begin{pmatrix} P_3 \\ Q_3 \end{pmatrix} = \begin{pmatrix} c' & d' \end{pmatrix} A \begin{pmatrix} \phi'(S) \\ \phi'(T + [r]S) \end{pmatrix} \tag{4.11}$$

from the simulator in the proof of Theorem 4.4. In the case of an honest prover (or a DMSIDH instance from $\mathcal{D}_0$ where $T' = \phi'(T)$), r here would be zero. Then,

$$\begin{aligned} R_1 = \widehat{\psi}'(R_3) &= \begin{pmatrix} c' & d' \end{pmatrix} A \begin{pmatrix} \widehat{\psi}'(\phi'(S)) \\ \widehat{\psi}'(\phi'(T + [r]S)) \end{pmatrix} \\ &= \begin{pmatrix} c' & d' \end{pmatrix} A \begin{pmatrix} \mathcal{O}_{E_0} \\ \widehat{\psi}'(\phi'(T)) \end{pmatrix} \end{aligned} \tag{4.12}$$

because again, $\phi'(S)$ is in the kernel of $\widehat{\psi}'$. Hence, we must have that $R_1 = \phi(R_0)$ regardless of which of the two distributions the DMSIDH instance was chosen from (and equivalently, which of the two Double-DMSIDH distributions). This implies that the coefficients a_i', b_i' in each SIDH square of the protocol exist and can be used to satisfy the verification algorithm regardless of whether a simulator or an honest prover generated the transcript.

Indistinguishability of the simulator: Suppose $\mathcal{B}_0$ is a PPT adversary which can distinguish between the simulation and the real transcripts for $\mathsf{chall} = 0$ with advantage Adv_0. We show that $\mathcal{B}_0$ can then also solve the Double-DMSIDH problem with the same advantage Adv_0. Let $(\psi_i, \psi_i', S_i, T_i, \phi'(S_i), T_i')_{i\in\{0,1\}}$ be an instance of the Double-DMSIDH problem. For both $i \in \{0,1\}$, we proceed as in the proof of Theorem 4.4 to create a transcript $\mathsf{com} = (E_{2,i}, E_{3,i}, P_{3,i}, Q_{3,i})_{i\in\{0,1\}}$, $\mathsf{chall} = 0$, $\mathsf{resp} = (c_i, d_i)_{i\in\{0,1\}}$. We also compute the images $U_i', V_i' = \psi_i(U), \psi_i(V)$ of the random basis (U, V), exactly as above. We then provide this transcript to $\mathcal{B}_0$. This will produce transcripts in a distribution identical to those produced by the simulator, because the steps are the same. Therefore, the response from $\mathcal{B}_0$ will solve the Double-DMSIDH problem with advantage Adv_0.

Now, coming to the $\mathsf{chall} = 1$ case, we similarly suppose $\mathcal{B}_1$ is a PPT adversary which can distinguish between the simulation and the real transcripts for $\mathsf{chall} = 1$ with advantage Adv_1. Let $(E_{2,i}, E_{3,i}, \phi_i', U_i', V_i')_{i\in\{0,1\}}$ be an instance of the Double-DSSP problem. As in the proof of Theorem 4.4, compute $P_{2,i}, Q_{2,i} \leftarrow \mathsf{CanonicalBasis}_2(E_{2,i})$, and let $P_{3,i}, Q_{3,i} := \phi_i'(P_{2,i}), \phi_i'(Q_{2,i})$. Finally, write $\ker(\phi_i') = \langle [e]U_i' + [f]V_i' \rangle$ and set $\mathsf{com} := (E_{2,i}, E_{3,i}, P_{3,i}, Q_{3,i}, U_i', V_i')_{i\in\{0,1\}}$, $\mathsf{chall} := 1$, and $\mathsf{resp} := (e, f)$, and give $(\mathsf{com}, \mathsf{chall}, \mathsf{resp})$ to $\mathcal{B}_1$. If $\mathcal{B}_1$ outputs 1, then we respond to the Double-DSSP instance with 1 and win with advantage Adv_1.

Recall that an adversary with non-negligible advantage against the Double-DMSIDH problem can solve the DMSIDH problem, by Theorem 2.16. Hence, assuming the Double-DSSP and DMSIDH problems are hard, transcripts generated by the simulator are indistinguishable from honest transcripts generated as per the protocol in Figure 4.5.

2-special soundness: Suppose we obtain two accepting transcripts $(\mathsf{com}, 0, \mathsf{resp})$ and $(\mathsf{com}, 1, \mathsf{resp}')$. The secret isogeny corresponding to the public key $X = (E_1, P_1, Q_1)$ can be recovered as follows, hence $\mathsf{Extract}$ can extract a valid witness W for the statement X such that $(X, W) \in \mathcal{R}_{\mathsf{SIDH}}$. From such a pair of commitments and responses, for each of the two SIDH squares committed to in Figure 4.5, we can recover $\phi_i : E_0 \to E_1$ of degree $\ell_1^{e_1}$ by the proof of Theorem 4.4. Now,

$$\begin{aligned} \ker(\phi_0) &= \widehat{\psi}_0(\ker(\phi_0')) \\ &= \langle \widehat{\psi}_0([e]U_0 + [f]V_0) \rangle \\ &= \langle \widehat{\psi}_1([e]U_1 + [f]V_1) \rangle \\ &= \widehat{\psi}_1(\ker(\phi_1')) \\ &= \ker(\phi_1). \end{aligned} \tag{4.13}$$

Therefore, we recover the same isogeny $\phi_0 = \phi_1 = \phi$ from both squares. For each of these two squares $i \in \{0,1\}$, the verifier will choose an $R_{0,i}$ and also learn its image $R_{1,i}$ under ϕ. This follows from Lemma 1.8, with $S := [c']P_2 + [d']Q_2$.

Now, because the two $R_{0,i} = [a_i']P_0 + [b_i']Q_0$ are independent, $\langle R_{0,0}, R_{0,1} \rangle$ forms another basis for $\langle P_0, Q_0 \rangle = E_0[\ell_2^{e_2}]$, with change-of-basis matrix

$$B = \begin{pmatrix} a_0' & b_0' \\ a_1' & b_1' \end{pmatrix}. \tag{4.14}$$

We can then see (due to Line 22 of verification) that

$$\begin{pmatrix} R_{0,0} \\ R_{0,1} \end{pmatrix} = B \begin{pmatrix} P_0 \\ Q_0 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} R_{1,0} \\ R_{1,1} \end{pmatrix} = B \begin{pmatrix} P_1 \\ Q_1 \end{pmatrix}, \tag{4.15}$$

and from Lemma 1.8, we also have

$$\begin{pmatrix} \phi(R_{0,0}) \\ \phi(R_{0,1}) \end{pmatrix} = \begin{pmatrix} R_{1,0} \\ R_{1,1} \end{pmatrix} = B \begin{pmatrix} \phi(P_0) \\ \phi(Q_0) \end{pmatrix}. \tag{4.16}$$

Therefore,

$$B \begin{pmatrix} \phi(P_0) \\ \phi(Q_0) \end{pmatrix} = B \begin{pmatrix} P_1 \\ Q_1 \end{pmatrix}, \tag{4.17}$$

and since B is invertible, we must have that $P_1 = \phi(P_0)$ and $Q_1 = \phi(Q_0)$, as required. □

Note that the protocol in Figure 4.5 essentially runs the previous protocol (in Figure 4.4) twice, while introducing the extra points U, V, hence the transcripts produced by this Proof of Knowledge for $\mathcal{R}_{\mathsf{SIDH}}$ will be more than twice the size.

Remark 4.8. Ghantous et al. [GPV21] discuss issues with extraction of a witness in two different scenarios. Their first scenario ("single collision") involves two distinct isogenies $\phi' : E_2 \to E_3$ in the SIDH square of the identification scheme. Neither of our new schemes are impacted by such collisions because the provision of points $P_3, Q_3 \in E_3$ uniquely determines the isogeny ϕ', as shown by Martindale and Panny [MP19] (c.f. Theorem 2.11). Their second scenario ("double collision") involves two distinct (non-equivalent) isogenies $\phi, \tilde{\phi} : E_0 \to E_1$, both of degree $\ell_1^{e_1}$ and a point $R \in E_0$ such that

$$E_1/\langle \phi(R) \rangle \cong E_1/\langle \tilde{\phi}(R) \rangle.$$

Our second protocol, for the relation $\mathcal{R}_{\mathsf{SIDH}}$, ensures that the witness extracted is a valid witness for the public key used (including the torsion points). Hence, this second collision scenario does not have any impact on the soundness of our protocol either. Again, the witness extracted will be unique, by Martindale and Panny [MP19].

4.4.1 Ideas for potential improvements

Figure 4.5 has each round commit and respond to two SIDH squares (giving 2κ SIDH squares in total). This is the basic requirement to ensure the prover can indeed perform the protocol using independent isogenies ψ with overwhelming probability, and admits a simple proof of 2-special soundness. However, it may be possible to do better than this. We shall discuss some ideas here.

The points U, V of order $\ell_1^{e_1}$ are used in the protocol for $\mathcal{R}_{\mathsf{SIDH}}$ to ensure that the $\phi' : E_2 \to E_3$ in each SIDH square are both derived from the same secret $\phi : E_0 \to E_1$. This problem arises because there may be multiple isogenies between E_0 and E_1 of degree $\ell_1^{e_1}$. An obvious example occurs by pre-composing ϕ with a non-trivial automorphism on E_0 with j-invariant 0, as we have already seen in Section 3.2.

If we make an assumption that the number of distinct isogenies of degree $\ell_1^{e_1}$ between the two curves is bounded (or at least, that it is infeasible to find more than a certain number of them), then we may be able to argue that using a sufficient number of rounds with different, "random enough" isogenies ψ is enough to prove existence and knowledge of ϕ. However, it is not clear how an extractor would work in this case. Reasoning about collisions is highly related to the work of Ghantous et al. [GPV21].

We now sketch an idea for a very different alternative protocol, though. As we have seen, the main idea of the proof for $\mathcal{R}_{\mathsf{SIDH}}$ is to require the prover to successfully respond with independent choices of ψ. Figure 4.5 does this with two independent pairs each round of the sigma protocol. We can go further than just two, though. Recall from Chapter 1 that there are $\ell_2 + 1$ different order-ℓ_2 subgroups of the ℓ_2-torsion on an elliptic curve E. If we write $\psi = [a]P_0 + [b]Q_0$ and consider the pair (a, b), this corresponds to the $\ell_2 + 1$ distinct independent classes of pairs a, b where two pairs (a_0, b_0) and (a_1, b_1) belong to the same class if $\lambda(a_1, b_1) \equiv (a_0, b_0) \pmod{\ell_2}$ for some invertible λ. Away from curves with j-invariant $0, 1728$, these $\ell_2 + 1$ independent classes correspond to the distinct "directions" you can leave E via isogenies of degree ℓ_2 (since the supersingular isogeny graph is $\ell_2 + 1$ regular). Obviously when $j(E) = 0$ or 1728, this idea of distinct directions no longer holds, but the number of distinct conjugacy classes remains the same. Whenever two such pairs are not conjugate, it holds that $a_0b_1 - a_1b_0$ is invertible (by basic properties of matrices), and thus that ψ_0 and ψ_1 are independent. Let $cl_0, \ldots, cl_{\ell_2}$ represent the $\ell_2 + 1$ conjugacy classes of (a, b) values, in a fixed ordering.

The proof will involve $(\ell_2+1)\lceil\kappa/\ell_2\rceil$ rounds of a sigma protocol similar to the one in Figure 4.4. In the first $\lceil\kappa/\ell_2\rceil$ rounds, the prover will only choose isogenies ψ such that if $\ker(\psi) = \langle [a]P_0+[b]Q_0\rangle$, then (a, b) lie in cl_0. Similarly, the second "block" of $\lceil\kappa/\ell_2\rceil$ sigma protocol rounds will all use isogenies ψ such that (a, b) lie in cl_1, and likewise for all subsequent blocks. The sigma protocol otherwise proceeds as in Figure 4.4 for commitment, challenge, and response.

For verification, for any round i in which $\mathsf{chall}_i = 0$, the verifier will recover ψ and ensure that the a, b pair generating its kernel is in the appropriate conjugacy class cl_j, where $j = \lfloor i\ell_2/\kappa\rfloor$. If this does not hold for any i, verification will fail. Then, verification of each round of the sigma protocol individually proceeds as in Figure 4.6, ignoring the for-loops, the checks on U_i, V_i, and the $a_0'b_1' - a_1'b_0'$ check (as each round only uses a single SIDH square now).

We have been unable to formally argue the soundness of this scheme, but intuitively, we believe it can offer κ-bit security after only κ rounds. This is because being able to consistently respond in two different "blocks" implies the prover is behaving honestly, as we have already seen. Due to requiring valid commitments and responses for pairs of challenges in different blocks, this protocol would have 4-special or some more general type of soundness, rather than 2-special soundness.

Precisely, if the prover can answer both $\mathsf{chall}_i = 0$ and $\mathsf{chall}_i = 1$ for some i in two (or more) different blocks, then they must be able to respond correctly using at least two independent isogenies ψ (or conjugacy classes of (a, b)). This immediately implies the correctness of the points in the public key, by the argument in Section 4.4. Thus, a prover can only reliably cheat in at most one of the $\ell_2 + 1$ blocks—leaving ℓ_2 blocks where a dishonest prover would need to guess the challenge in order to cheat (otherwise they would be caught with probability 1/2 in each of the remaining $\ell_2\lceil\kappa/\ell_2\rceil$ rounds). Because $(\ell_2 + 1)/\ell_2 < 2$ for all primes ℓ_2, and the points U_i', V_i' are no longer needed in the commitments, this is an improvement in transcript size over the double square protocol in Section 4.4. For example, this would give only $4\kappa/3$ rounds in the case $\ell_2 = 3$.

The fact that the prover must use multiple different isogenies ψ in all conjugacy classes cl_i also appears to limit their ability to exploit "collisions" (multiple isogenies of the same degree between curves), as mentioned above. This protocol is therefore interesting and worthy of further thought and analysis, but we leave that for future work. It is likely that other improvements to the efficiency and size of the schemes we presented in this chapter are also possible with more analysis.

4.5 SIDH signatures and non-interactive zk-PoKs

We conclude with some brief, standard remarks about the use of the new zero-knowledge proof protocols we have proposed in this chapter.

It is standard to construct a non-interactive signature scheme from an interactive protocol using the Fiat–Shamir transformation (secure in the (quantum) random oracle model [LZ19]). This works by making the challenge chall for the t rounds of the ID scheme a random-oracle output, from input the vector of commitments $\mathsf{com} = [\mathsf{com}_i]_{i \in \{1,\dots,t\}}$ and a message M. That is, for message M,

$$V_1^{\mathcal{O}}(\mathsf{com}) := \mathcal{O}(\mathsf{com} \parallel M). \tag{4.18}$$

Thus, the prover does not need to interact with a verifier and can compute a non-interactive transcript. This is a standard design pattern for signature schemes, because the transcript now also commits to the message M.

Because the sigma protocol described in the preceding sections not only proves knowledge of the secret isogeny between two curves, but also correctness of the torsion points in the public key, we obtain a signature scheme that is also a Proof of Knowledge of the secret key corresponding to a given SIDH public key. As we have discussed, such a proof guarantees that the SIDH public key is well-formed, since a valid secret key (isogeny) must exist. For example, simply signing a public key (as the message M) with its own secret key using the new scheme gives a simple NIZK proof of well-formedness for the public key, which provides protection against adaptive attacks.

The unforgeability of such a scheme is additionally based on the CSSI problem (Definition 2.2). This is because 2-special soundness of the sigma protocols only guarantees that, given two transcripts for the same commitment but with different challenges, we can recover the secret isogeny $\phi : E_0 \to E_1$. It does not guarantee that recovering ϕ from E_0 and E_1 alone (or from an SIDH public key) is difficult. Assuming the CSSI problem is hard, though, we achieve the desired result. This part of the proofs given in the literature (for example, by Galbraith et al. [GPS20]) remains unchanged by the results we have presented in this chapter.

Such a NIZK Proof of Knowledge of an SIDH secret key can, among other applications, be used to achieve a secure non-interactive key exchange scheme based on SIDH. Specifically, it would enable both participants to verify non-interactively that the other participant's key is honestly formed and safe to use without fear of adaptive attack. We will see this done in Chapter 5.

Chapter 5

Post-Quantum Signal Key Agreement with SIDH

Signal is a widely-used secure messaging protocol with implementations in its namesake app (Signal Private Messenger), as well as others including WhatsApp, Facebook Messenger and more. Due to its popularity, it is an interesting problem to design a post-quantum secure variant of the protocol. However, some difficulty arises due to the lack of a formally-defined security model or properties for the original protocol itself.

The Signal protocol consists of two general stages: the first is the initial key agreement, which is then followed by the double ratchet protocol [MP16a]. The initial key agreement is currently done via a protocol known as Extended Triple Diffie–Hellman (X3DH) [MP16b]. While Alwen, Coretti, and Dodis [ACD19] construct a version of the double ratchet component using key encapsulation mechanisms (KEMs), which can be made post-quantum secure, the X3DH stage has proven to be more subtle and challenging to replace in an efficient way with post-quantum solutions. Recent work by Brendel, Fischlin, Günther, Janson, and Stebila [BFG$^+$20] examines some of these challenges and suggests that SIDH cannot be used to make X3DH post-quantum secure due to its vulnerability to adaptive attacks when static keys are used.

Specifically, Brendel et al. are referring to the adaptive attack described in Section 3.1, known as the GPST attack. The Signal X3DH protocol is an authenticated key exchange (AKE) protocol, requiring keys from both parties involved. Without a secure method of validating the other party's public key correctness, it would be insecure to perform a naive SIDH key exchange. For example, the initiator of a key exchange could adaptively modify the ephemeral public keys they use to learn the receiver's long-term identity private key, as we have seen. None of the previous methods of validation that we examined in Chapter 3 are well-suited to solving this issue in the Signal X3DH context. We have already observed that the Weil pairing is not sufficient to detect adaptive attacks, and using k-SIDH [AJL17] would be very inefficient. While the Fujisaki–Okamoto (FO) transformation (Section 3.4) is able to prevent adaptive attacks in some cases, it cannot handle scenarios where both parties use static identity keys. We cannot exclude the possibility that the long-term keys are used as part of an attack: a dedicated or well-resourced attacker could certainly register many new accounts whose identity keys are maliciously crafted, and initiate exchanges with an unsuspecting user (perhaps by marauding as their friends or colleagues) to learn their secret key. For these reasons, Brendel et al. disregard SIDH as a contender and suggest using CSIDH [CLM$^+$18] for an isogeny-based variant of Signal. However, this primitive is much less efficient than SIDH—in part due to sub-exponential quantum attacks that lead to much larger parameters.

One of the primary goals of this chapter is to show that SIDH can indeed be used to construct a post-quantum X3DH replacement that satisfies the same security model as the original X3DH protocol—despite the claim by Brendel et al. [BFG$^+$20].

In order to design good post-quantum replacements for the Signal protocol, a clear security model is required. This is an area of difficulty because the original Signal protocol did not define a security model—it appears to be designed empirically. There have since been a few efforts to formalise the security properties of the Signal protocol and X3DH. Notably, the work by Cohn-Gordon, Cremers, Dowling, Garratt, and Stebila [CCD$^+$20] was the first to propose a security model and prove the security of Signal in it. The recent work of Hashimoto, Katsumata, Kwiatkowski, and Prest [HKKP21] also proposes a generic security model for the Signal initial key agreement (specifically, for what they call Signal-conforming AKEs), and gives a generic construction from KEMs and signature schemes (as mentioned above, KEMs do not allow static–static key exchange, so a signature scheme is required to provide explicit authentication of the initiating party). From these analyses of the protocol, the following security properties have been identified as important, which any post-quantum replacement should therefore also satisfy:

1. Correctness: If Alice and Bob complete an exchange together, they should derive the same shared secret key.
2. Secrecy (also known as key-indistinguishability): Under the corruption of various combinations of the participants' secret keys, the shared secret for the session should not be recoverable, or even distinguishable from a random key. The combinations are defined by the specific security model used, for example, the CK model [CK01] or the model in [CCD$^+$20]. This is, of course, a basic requirement for any secure key exchange.
3. (Implicit) authentication: Both participants should know who they are talking to, and be able to verify their partner's identity.
4. Perfect forward secrecy (PFS): Past communication should remain secure and unreadable by adversaries even if the participants' long-term keys are compromised in the future.
5. Asynchronicity: The protocol can be made non-interactive by hosting participants' public keys on a third-party server, which is untrusted. In the security model, the only possible malicious ability the server should have is that it could deny Alice the retrieval of Bob's keys (or, say, not give out his one-time keys). This property is also called **receiver obliviousness** by Hashimoto et al. [HKKP21], because the uploaded public keys are not intended for any particular user, but can be retrieved and used by anyone.
6. (Offline) deniability [VGIK20], also known as identity-hiding: The transcript of an exchange session should not reveal the participants of the exchange (in a non-repudiable way).

We propose a new, efficient, post-quantum key exchange protocol using SIDH, modelled after X3DH, which we call SI-X3DH. This new protocol solves the problem of adaptive attacks by using a variant of the FO transformation to prove that the initiator's ephemeral key is honestly generated, and the SIDH Proof of Knowledge (from Chapter 4) to prove each party's long-term keys are well-formed—something which only needs to be verified once (and could be offloaded to the PKI server depending on the trust model used). We prove security of the SI-X3DH protocol formally in a new model defined in Section 5.3, in the random oracle model (ROM). The security proof reduces to the hardness of the VCDH and HCDH problems we defined in Section 2.3.2, which we showed further reduce to the hardness of the SI-CDH assumption in the ROM. In that respect, the security of the SI-X3DH protocol reduces to standard isogeny hardness assumptions.

Because SIDH is an efficient post-quantum key exchange proposal with very small key sizes (although still larger than classical elliptic curve keys used in the original X3DH), and because SI-X3DH requires only three or four SIDH exchanges (unlike k-SIDH), our protocol is also efficient and practical. For example, SIDH is much faster than CSIDH—suggested in the proposal by Brendel et al. [BFG$^+$20]—because CSIDH uses larger-prime degree isogenies while SIDH commonly uses only isogenies of degree (a power of) two and three. Our scheme also does not rely on expensive machinery such as post-quantum ring signatures to achieve deniability (as [HKKP21] does). However, a large drawback of our scheme is that it relies on proving knowledge of the secret long-term identity keys, by using the protocol we proposed in Chapter 4 for example. This only needs to be done once per contact (or could be offloaded to the keyserver, depending on the trust model), but for users who add many new contacts regularly, this may create an unacceptable overhead. The efficiency of our scheme is discussed more in Section 5.6.

Another disadvantage of our scheme, as discussed in Section 5.4, is that SI-X3DH suffers from the possibility of more permanent key compromise impersonation (KCI) than the original Signal X3DH protocol does. Technically, neither Signal X3DH nor SI-X3DH satisfy the KCI resistance requirement of the eCK and CK+ security models, but there is a practical difference between the schemes. Impersonation was possible with the compromise of the semi-static key in Signal X3DH, whereas in SI-X3DH, impersonation is possible with compromise of the long-term identity key. Thus, cycling the semi-static key is no longer sufficient to prevent long-term impersonation. This is worth considering, but we believe the change is acceptable, as medium-term impersonation seems just as damaging as long-term, and corruption of an identity key is a severe break in security anyway.

We begin with an overview of recent work on post-quantum Signal X3DH replacement proposals, in order to understand the improvements our scheme makes. We then recall the existing Signal X3DH scheme in Section 5.2, and discuss an appropriate security model for it in Section 5.3—this includes the definition of our new security model, the Signal-adapted-CK model. Subsequently, we present the construction of our new SI-X3DH protocol in Section 5.4 using SIDH, and give a proof of security for it in Section 5.5. Finally, Section 5.6 discusses the efficiency of our protocol and the key differences between SI-X3DH and the original X3DH scheme.

5.1 Relation to other work

As we will soon see, the SI-X3DH protocol we propose has some structural differences from X3DH. In particular, SI-X3DH performs an SIDH exchange between the two parties' identity keys (IK_A and IK_B), whereas previously, X3DH used IK_A and SK_B instead (involving Bob's semi-static key, rather than his identity key). Due to the asymmetry between the degrees of the isogenies the two parties in SIDH use, our protocol requires parties to register two keys rather than one: a receiving key and a sending key. Finally, in order to prevent adaptive attacks, SI-X3DH uses a single FO-proof per exchange, and a once-off proof of well-formedness of each party's identity keys (see Section 5.4 for discussion of this). Despite these differences, the structure of the protocol more closely resembles X3DH than any of the other post-quantum proposals presented to date. For example, our protocol allows Bob the balance between one-time keys and medium-term (semi-static) keys—where the former may be exhausted, leading to denial of service, while the latter provide less security in some attack scenarios. These properties and differences are discussed further in Section 5.3.

The original Signal X3DH scheme requires Bob to sign his semi-static keys, to prevent a malicious keyserver from providing its own keys and compromising the perfect forward secrecy guarantee of the scheme. This requirement must still hold in any post-quantum replacement too (although some literature omits reference to it). In general, these X3DH replacements (including SI-X3DH) are agnostic to the signature scheme used for this purpose, so any efficient post-quantum signature scheme may be used alongside them—there is no restriction to use an isogeny-based signature scheme with SI-X3DH. Regardless of the choice, a signature verification key must be registered independently of the public keys used in the exchange itself.

Brendel et al. [BFG$^+$20] proposed a new model for post-quantum X3DH replacements using a primitive they call split-KEMs. Their construction is a theoretical work, as they leave it an open question whether post-quantum primitives such as CSIDH satisfy the security definitions of their split-KEM.

Recently, Hashimoto et al. [HKKP21] presented their Signal-Conforming AKE (SC-AKE) construction, also using post-quantum KEMs to construct a generic Signal X3DH replacement. To achieve deniability, their scheme requires a post-quantum ring signature scheme. Independently, but following a very similar approach to Hashimoto et al., Brendel et al. [BFG$^+$22] also proposed a deniable AKE using post-quantum KEMs (which they call "Signal in a Post-Quantum Regime" (SPQR)) and a designated verifier signature (DVS) scheme. As they mention, little work has been done to date in constructing DVS schemes from post-quantum assumptions, so Brendel et al. also propose using a two-party post-quantum ring signature scheme for the same purpose.

We briefly outline the differences between these works and that presented in this chapter using Table 5.1, with the original Signal X3DH protocol included as a reference.

Scheme	PQ-secure	Deniable	Requires sig	Long-term data	Exchanged data
Original Signal X3DH protocol	×	✓	✓	K	3 keys
Split-KEM based X3DH [BFG$^+$20]	✓	?	✓	K, K_σ	3 keys, 4 ciphertexts
Signal-Conforming AKE [HKKP21]	✓	*with PQ ring signature	✓(×2)	K, K_σ, K_σ^*	1 key, 3 ciphertexts
SPQR [BFG$^+$22]	✓	*with PQ ring signature or DVS	✓(×2)	K, K_σ, K_σ^*	2 keys, 4 ciphertexts
SI-X3DH (this chapter)	✓	✓	✓	K_2, K_3, K_σ + PoK	3 keys, 1 ciphertext

Table 5.1: Comparison of post-quantum Signal X3DH replacements. *Long-term data* refers to the size of the initial registration cost for each user (the "offline" data). *Exchanged data* gives the amount of ephemeral data sent in a single exchange (by both parties combined), that is, the size of the "online" transcript. Note that all schemes require a signature scheme (*Requires sig*) to obtain PFS—post-quantum schemes use a separate signature verification key K_σ while Signal X3DH reused the same key K for both exchange and signature verification (with ECDH and XEdDSA [Per16]).

The Split-KEM protocol [BFG$^+$20] does not discuss the requirement for a signature scheme on the semi-static keys, but the same attack on PFS applies to their scheme as it does to the original Signal X3DH protocol if the semi-static keys are not signed—a malicious server or tampering man-in-the-middle can insert their own semi-static key rather than Bob's, and later compromise Bob's long-term identity key, thus allowing recovery of the shared secret. The Signal-Conforming AKE protocol and SPQR protocol require this signature for PFS too, for the same reason. These latter two schemes also use a second (ring/DVS) signature (discussed below)—two signatures per exchange. Because ring signatures and DVS schemes are much more expensive than standard signatures, for efficiency it would likely be preferable to use two separate schemes, hence the two signing keys K_σ, K_σ^* in Table 5.1. Our construction, as mentioned above, requires a single signature on the semi-static key. Because there are no efficient post-quantum constructions with a public key that can be used in both a signature scheme and a key exchange, requiring a separate signature scheme (and verification key) seems unavoidable for any post-quantum X3DH replacement.

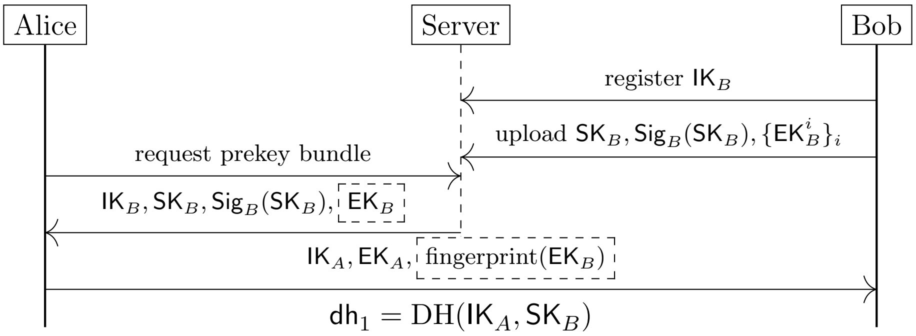
For deniability, SC-AKE requires the initiator of the key exchange to sign the session ID. This signature creates non-repudiable evidence of the initiator's involvement in the exchange. Hashimoto et al. [HKKP21] and Brendel et al. [BFG$^+$22] suggest using a ring signature to attain deniability. Specifically, a signature under a two-party ring involving just the sender and receiver is sufficient to authenticate the other party in the exchange (since one party knows the signatures that they themselves generated), but to a third party, the signature could have been generated by either participant. Unfortunately, however, a post-quantum ring signature scheme is a much more expensive construction than a standard signature. Deniability of the split-KEM construction is not discussed by the 2020 work of Brendel et al. [BFG$^+$20], and would appear to depend on how the split-KEM is instantiated. We emphasise that the signature on Bob's semi-static keys mentioned above does not have any impact on deniability, as that signature exists independently of any particular exchange session or counterparty. These deniability drawbacks are only caused by signatures on session-specific information like the session ID, for the sake of authentication.

Finally, it is important to note that the SC-AKE protocol does not use a semi-static key—only long-term and ephemeral keys. This means that unlike in Signal X3DH, if a receiver is offline for an extended period of time, it is possible for all the ephemeral keys they uploaded to the server to be exhausted (either due to popularity or a malicious attempt to do so). This creates an opportunity for denial of service which is not present when semi-static keys are used and the ephemeral component is optional. Brendel et al. [BFG$^+$22] address this by using a semi-static and an ephemeral KEM encapsulation key if available, as in Signal's X3DH.

In other recent work, Fouotsa and Petit [FP21] propose a protocol similar to SIDH which they claim is not vulnerable to adaptive attacks. They call this protocol HealSIDH ("healed" SIDH). This protocol operates by requiring participants to also reveal the action of their isogenies on points of larger order than in SIDH. However, this protocol is interactive and would not allow a key exchange to take place while one participant is offline—it requires the receiver to send certain points back to the initiator for validation before the exchange can be completed. Specifically, this fails the requirement of asynchronicity, so would not be suitable for use in a Signal X3DH replacement. It is for the same reason that proposals for post-quantum TLS handshake replacements, including by Schwabe, Stebila, and Wiggers [SSW20], also fail to be applicable in the Signal context—these protocols involve messages sent by both parties sequentially over multiple rounds, and often do not authenticate one of the two parties (the client).

5.2 The Signal X3DH protocol

The basic process of the X3DH protocol is given in Figure 5.1, where Alice is the initiator and Bob is the responder. Let $\mathrm{DH}_{\mathsf{pp}}(A, B) = g^{ab} \pmod N$ denote the result of a Diffie–Hellman key exchange between keys $A = g^a$ and $B = g^b$ (at least one of the private keys a, b is needed to compute this, but the result is unambiguous), with public parameters pp including g and N. Because we assume fixed public parameters, we will usually omit the subscript pp. We remind


Alice
Server
Bob
register IK_B
upload $\mathsf{SK}_B, \mathsf{Sig}_B(\mathsf{SK}_B), \{\mathsf{EK}_B^i\}_i$
request prekey bundle
$\mathsf{IK}_B, \mathsf{SK}_B, \mathsf{Sig}_B(\mathsf{SK}_B), \boxed{\mathsf{EK}_B}$
$\mathsf{IK}_A, \mathsf{EK}_A, \boxed{\text{fingerprint}(\mathsf{EK}_B)}$

$$\mathsf{dh}_1 = \mathrm{DH}(\mathsf{IK}_A, \mathsf{SK}_B)$$
$$\mathsf{dh}_2 = \mathrm{DH}(\mathsf{EK}_A, \mathsf{IK}_B)$$
$$\mathsf{dh}_3 = \mathrm{DH}(\mathsf{EK}_A, \mathsf{SK}_B)$$
$$\boxed{\mathsf{dh}_4 = \mathrm{DH}(\mathsf{EK}_A, \mathsf{EK}_B)}$$
$$K = \mathrm{KDF}(\mathsf{dh}_1 \parallel \mathsf{dh}_2 \parallel \mathsf{dh}_3 \boxed{\parallel \mathsf{dh}_4})$$

Figure 5.1: The X3DH protocol [MP16b]. dh_4 is optional on the basis of one-time key availability.

the reader that dashed boxes are used in this thesis to denote optional parameters which may be omitted.

Because the X3DH protocol is designed to work when the recipient (Bob) is offline, Alice obtains his public key information from a server. IK_A and IK_B are the fixed long-term identity keys of Alice and Bob respectively. Bob additionally uploads a semi-static public key SK_B signed by him to the server, which he rotates semi-regularly. He also uploads a number of one-time keys EK_B, but the use of these is optional, as the supply on the server may run out.

After Alice has received Bob's identity, semi-static, and (optional) one-time keys from the server, she performs a three- or four-part key exchange with her own identity key and ephemeral key. The three or four intermediary shared values computed are specified in the figure (denoted by dh_i), and are combined using some sort of secure hash or key derivation function (KDF). We shall assume they are simply concatenated and hashed with a cryptographic hash function. This results in the master shared secret for the exchange, which is then used in subsequent protocols such as Signal's Double Ratchet protocol.

Finally, Alice sends to Bob identifiers of which of his semi-static and one-time public keys she used (for example, short fingerprints), as well as her own identity and ephemeral keys. This allows Bob to also compute the same shared master secret.

Verification of the long-term identity keys is out-of-scope for the protocol and may be done either by trusting a third party (e.g. the server) as a PKI, or verifying the keys in-person or out-of-band in some other way.

5.3 Security model

Authenticated key exchange (AKE) security is a complex field of properties and models. Of primary interest is the notion of key indistinguishability, sometimes simply known as AKE security due to its universality. The seminal work by Bellare and Rogaway [BR93] defined a security model for authenticated key exchange (known as the BR model). Security in the BR model is based on the indistinguishability of true session keys from random, even when the adversary is given certain powers to control protocol flow, interactions, and to reveal long-term secret keys and states. A number of other models have since been developed, based on this original BR model, including the CK model [CK01] (named after its creators, Canetti and Krawczyk), the CK+ model [Kra05], and the eCK model [LLM07]. These models all differ in the specific powers the adversary is granted in the key-indistinguishability game (as well as having other differences such as how partner sessions and session IDs are defined). The main difference between the CK/CK+ models and the eCK model is that the latter uses ephemeral-key reveal queries while the former two use session-state reveal queries. These models are incomparable, as shown by Cremers [Cre09].

The eCK and CK+ models are generally viewed as the strongest and most desirable models, as they capture attacks that are outside the scope of the CK model: weak perfect forward secrecy (wPFS), key compromise impersonation (KCI), and maximal exposure (MEX). All of these properties relate to certain combinations of long-term and ephemeral keys being compromised by an adversary. Security in these models relies on allowing the adversary all non-trivial combinations of exposure—that is, any combination of keys from both parties that does *not* form a vertex cover on the graph of Diffie–Hellman exchanges in the protocol (the graph whose nodes are keys, and edges represent that a DH key exchange between the two incident keys is used in the protocol). A vertex cover would trivially allow the adversary to compute the shared secret, because in that case, at least one secret is known to the adversary in every DH exchange (edge). However, if the adversary does not have a vertex cover, at least one DH exchange (edge) cannot be computed, because the adversary does not have either of the secret keys involved. In this case, the overall session key of the protocol should remain hidden. We refer the reader to the work of Fujioka, Suzuki, Xagawa, and Yoneyama [FSXY12] for a more detailed analysis of the difference between these models.

Unfortunately, Signal X3DH does not meet the definition of security required by all of these models. This was observed by Cohn-Gordon et al. [CCD^{+}20]. Precisely, there do not exist edges in the exchange graph for every possible pair of keys—for example, there is no DH exchange between Alice's identity key and Bob's identity or ephemeral keys. Our benchmark for security is that a replacement protocol should meet at least the same security definition as that of the original protocol, so we must observe where exactly the original protocol breaks down in the eCK model. This allows us to propose a slightly weaker model, though still stronger than the CK model, that successfully represents the security goals of Signal X3DH. This gives a more formal and well-defined security model than the one Cohn-Gordon et al. [CCD^{+}20] used to prove security of the original Signal X3DH protocol. We call our new security model the Signal-adapted-CK model.

The recent work of Hashimoto et al. [HKKP21] provided a similar security model, for what they call a Signal-conforming AKE protocol. Their security model differs from ours in the fact that it does not take semi-static keys into account (their proposed construction does not use semi-static keys). They also use the language of state-reveals rather than ephemeral-key-reveals. Their model is stronger than the Signal-adapted-CK model—in fact, the original Signal X3DH protocol would not satisfy their model (it requires security against the two events E_4 and E_8 in Table 5.3, discussed further below). However, our goal is to propose a model that exactly captures the security properties of the original Signal X3DH protocol, which was not the goal of their model. In other words, we wish to analyse Signal, not some stronger protocol.

Before we begin, let us briefly recall the meanings of the security notions mentioned above:

- Perfect forward secrecy (PFS) implies that an adversary who corrupts one or both of the participants' long-term secret keys should not be able to reveal the session key of previous sessions executed by those participants—the past remains secure. This is achieved by the use of ephemeral keys whose corresponding secrets are erased on successful completion of the exchange protocol. *Weak* PFS implies that this PFS is only achieved if adversaries cannot interfere with the protocol during the exchange (e.g., man-in-the-middle attacks), they can only attack it after the fact.

- Key compromise impersonation (KCI) resistance captures the scenario where an adversary reveals or corrupts the long-term secret key of a participant A: the adversary should be unable to impersonate other parties *to* A (but of course, can still impersonate A to other parties). For example, if Carol has compromised Alice's secret keys, she should be unable to send messages to Alice that Alice believes came from an uncorrupted third party, Bob.
- The maximal exposure (MEX) property states that, when given any one (long-term or ephemeral) secret key of each party in an exchange, the adversary should still be unable to distinguish the real session key from random. This property essentially takes into account all other combinations of keys that may be compromised in practice, hence the "maximal" denomination.

Standard security models generally define keys to be either long-term or ephemeral. As a recipient in the Signal protocol uses up to three keys, including a semi-static (medium-term) key, it is not at first obvious how to integrate this semi-static key into such two-key models. We choose to consider it as both long-term and ephemeral in different situations. This is discussed further in Remark 5.3.

We define the formal key indistinguishability experiment for our Signal-adapted-CK model now. We then provide a proof of security of our construction in this model in Section 5.5.

5.3.1 Key indistinguishability experiment

Let $\mathcal{K}$ denote the space of all possible session keys that could be derived in an exchange between two parties. We model n parties $P_1, \dots, P_n$ through oracles Π_i^j, denoting the j-th session run by participant P_i. We limit the number of sessions per party by $1 \leq j \leq S$. Each oracle has access to the secret key of the corresponding party P_i's fixed long-term identity key IK_i, as well as the secrets for each of the m semi-static keys $\mathsf{SK}_i^1, \dots, \mathsf{SK}_i^m$. Each oracle also has the following local variables:

- $\Pi_i^j.\mathsf{rand}$: The fixed randomness of oracle i for its j-th session (where Π_i^j is deterministic based on this randomness).
- $\Pi_i^j.\mathsf{role} \in \{\perp, \texttt{init}, \texttt{resp}\}$: The role of participant i in their j-th exchange.
- $\Pi_i^j.\mathsf{sk\_id}$: The index ℓ of the semi-static key SK_i^ℓ that participant i uses in their exchange j.
- $\Pi_i^j.\mathsf{peer\_id}$: The index k of the alleged peer P_k in the j-th exchange of oracle i.
- $\Pi_i^j.\mathsf{peer\_sk\_id}$: The index ℓ of the alleged peer's semi-static key $\mathsf{SK}_{\mathsf{peer\_id}}^\ell$ used in the exchange.
- $\Pi_i^j.\mathsf{sid}$: The session ID, explained further below.
- $\Pi_i^j.\mathsf{status} \in \{\perp, \texttt{accept}, \texttt{reject}\}$: Indicates whether the oracle has completed this session of the key exchange protocol and computed a session key from the exchange.
- $\Pi_i^j.\mathsf{session\_key} \in \mathcal{K}$: The computed session key.

These values are all initialised to $\perp$ at the start of the security experiment, except rand, which is initialised with random coins for each oracle. The oracle status is set to `accept` or `reject` on the computation of $\mathsf{session\_key}$.

The session ID is a feature of the security experiment, not the real protocol. We define the session ID to be a tuple $(\Pi, \mathsf{IK}_{\mathcal{I}}, \mathsf{IK}_{\mathcal{R}}, \mathsf{SK}_{\mathcal{R}}, \mathsf{EK}_{\mathcal{I}}, \boxed{\mathsf{EK}_{\mathcal{R}}})$ where $\mathcal{I}, \mathcal{R}$ denote the initiator and responder respectively, Π is a protocol identifier, and $\mathsf{EK}_{\mathcal{R}}$ is optional (so may be null). We say two sessions with the same sid are **matching**. This is done to restrict the adversary from making queries against any session matching the test session for the game—to avoid trivialising security. For a session Π_i^j we also define a **partner** session to be any session Π_k^ℓ for which $\Pi_i^j.\mathsf{peer\_id} = k$ and $\Pi_k^\ell.\mathsf{peer\_id} = i$, $\Pi_i^j.\mathsf{role} \neq \Pi_k^\ell.\mathsf{role}$, and $\Pi_i^j.\mathsf{sid} = \Pi_k^\ell.\mathsf{sid}$. We say any two such sessions are **partners**. Note that if two sessions are partners, they are also, by definition, matching.

Setup The security game is played between challenger $\mathcal{C}$ and a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$. $\mathcal{C}$ will generate identity keys for the n participants, $\mathsf{IK}_1, \ldots, \mathsf{IK}_n$, and for each participant i, generate m semi-static keys $\mathsf{SK}_i^1, \ldots, \mathsf{SK}_i^m$. $\mathcal{C}$ will finally choose a uniformly random secret bit $b \leftarrow \{0,1\}$, and provide $\mathcal{A}$ with access to the oracles Π_i^j.

Game Adversary $\mathcal{A}$ can adaptively make the following queries in the game:

- **Send**(i, j, μ): Send an arbitrary message μ to oracle Π_i^j. The oracle will behave according to the key exchange protocol and update its status appropriately.
- **RevealIK**(i): Return the secret long-term key of participant i. After this, participant i is **corrupted**.
- **RevealSK**(i, ℓ): Return the ℓ-th secret semi-static key of participant i. After this, SK_i^ℓ is said to be **revealed**.
- **RevealEK**(i, j): Return the ephemeral key (i.e., the random coins) of the j-th session of participant i. After this, EK_i^j and $\Pi_i^j.\mathsf{rand}$ are said to be **revealed**.
- **RevealSessionKey**(i, j): Return $\Pi_i^j.\mathsf{session\_key}$. After this, session Π_i^j is said to be **revealed**.

Test At some point in the game, $\mathcal{A}$ will issue a special **Test**(i, j) query exactly once. $\mathcal{C}$ will return K_b to the adversary, where $K_0 := \Pi_i^j.\mathsf{session\_key}$ and $K_1 \leftarrow \mathcal{K}$ (a random key from the keyspace). After this query is made, session Π_i^j is said to be **tested**. $\mathcal{A}$ can continue to adaptively make queries to the above game functions after the Test query has been issued. Finally, $\mathcal{A}$ outputs a bit $b^* \in \{0,1\}$ as their guess.

At this point, the tested session Π_i^j must be **fresh**. Freshness is defined in Definition 5.1, and the cases for freshness are also summarised in Table 5.2 for clarity. If the tested session is not fresh, $\mathcal{C}$ will abort the game and output a uniformly random bit $b' \leftarrow \{0,1\}$ on behalf of $\mathcal{A}$.

Definition 5.1 (Freshness)**.** A session Π_i^j, with $\Pi_i^j.\mathsf{peer\_id} = k$, is **fresh** if *none* of the following hold:

- $\Pi_i^j.\mathsf{status} \neq \mathtt{accept}$.
- The $\mathsf{session\_key}$ of Π_i^j, or any matching session, is **revealed**.
- If $\Pi_i^j.\mathsf{role} = \mathtt{init}$:
  - Both **RevealIK**(i) and **RevealEK**(i, j) are issued.

  - Π_i^j has a partner Π_k^ℓ for some ℓ, **RevealIK**(k) is issued, and either **RevealSK**$(k, \Pi_i^j.\mathsf{peer\_sk\_id})$ $(\star)$ or **RevealEK**(k, ℓ) are issued. See Remark 5.3.

- If $\Pi_i^j.\mathsf{role} = \mathtt{resp}$:
  - Π_i^j has a partner Π_k^ℓ for some ℓ and both **RevealIK**(k) and **RevealEK**(k, ℓ) are issued.
  - **RevealIK**(i) and either **RevealSK**$(i, \Pi_i^j.\mathsf{sk\_id})$ $(\star)$ or **RevealEK**(i, j) are issued. See Remark 5.3.
- Π_i^j has no partner session and **RevealIK**$(\Pi_i^j.\mathsf{peer\_id})$ is issued.

To define security in this model, we require correctness and soundness. Soundness ensures that, if the adversary is restricted to making only reveal queries that keep the test session **fresh**, then its advantage in distinguishing the session key from random is negligible.

Definition 5.2. Let $\mathcal{A}$ be a PPT adversary. We define the advantage of $\mathcal{A}$ in winning the above key indistinguishability experiment kie with n parties, m semi-static keys per party, and S sessions per party, as

$$\mathsf{Adv}^{\mathrm{kie}}_{n,m,S}(\mathcal{A}) = \left| \Pr\left[\, b = b^* \,\right] - \frac{1}{2} \right| .$$

An authenticated key exchange protocol Π is secure in the Signal-adapted-CK model if it is:

- **Correct**: Any two parties following the protocol honestly derive the same sid, $\mathsf{session\_key}$, and both arrive at an $\mathtt{accept}$ state.
- **Sound**: The advantage of any PPT adversary $\mathcal{A}$ is $\mathsf{Adv}^{\mathrm{kie}}_{n,m,S}(\mathcal{A}) \le \mathsf{negl}$.

We emphasise that Table 5.2 and our definition of freshness in Definition 5.1 are strictly weaker than the standard eCK/CK+ cases and definitions—specifically, we have removed the adversary's ability to perform two specific cases of KCI attack. Both of these removed cases are given in Table 5.3, and correspond to the extra restrictions on freshness marked with a $(\star)$ in Definition 5.1. These are the cases that weaken the eCK/CK+ models to our Signal-adapted-CK model.

The reason for these exclusions from our model is that the original Signal X3DH protocol does not satisfy these properties, and our goal is to precisely model the security of that original protocol. Hence, these cases should be removed. The KCI attack on the original protocol is as follows: if Bob's semi-static key SK_B is compromised, an adversary can impersonate anyone to Bob. This is because Alice is only authenticated through dh_1 (the exchange with SK_B), so an adversary can claim the use of any other public key IK_E and calculate the correct Diffie–Hellman value with SK_B. As SK_B is periodically replaced by Bob, the impersonation to Bob can last only as long as he accepts exchanges with that particular SK_B. However, we consider this a failure of the KCI property because SK_B is not ephemeral. This is discussed further in Remark 5.3.

Remark 5.3. In the original Signal X3DH protocol, the semi-static keys SK_B are used to strike a balance between perfect forward secrecy and key-exhaustion denial of service. To correctly model the purpose of this key, we assume it is "ephemeral enough" to have been replaced some time before a PFS attack (event E_5 in Table 5.2) takes place—this is generally a longer-term attack and the cycling of the semi-static key is designed to prevent this precise attack.

Event	Matching session exists	$\mathsf{IK}_{\mathcal{I}}$	$\mathsf{EK}_{\mathcal{I}}$	$\mathsf{IK}_{\mathcal{R}}$	$\mathsf{SK}_{\mathcal{R}}$	$\mathsf{EK}_{\mathcal{R}}$	Attack
E_1	No	✓	x	x	✓	-	KCI
E_2	No	x	✓	x	x*	-	MEX
E_3	No	x	-	x	x*	✓	MEX
E_5	Yes	✓	x	✓	x	x	wPFS
E_6	Yes	x	✓	x	x*	✓	MEX
E_7	Yes	✓	x	x	✓	✓	KCI

Table 5.2: Behaviour of the adversary in our model, corresponding to the various freshness conditions in Definition 5.1. $\mathcal{I}$ and $\mathcal{R}$ denote whether the key belongs to the initiator or responder respectively. "✓" means the corresponding secret key is revealed or corrupted, "x" means it is not revealed, and "-" means it does not exist or is provided by the adversary.
*Discussed further in Remark 5.3.

Event	Matching session exists	$\mathsf{IK}_{\mathcal{I}}$	$\mathsf{EK}_{\mathcal{I}}$	$\mathsf{IK}_{\mathcal{R}}$	$\mathsf{SK}_{\mathcal{R}}$	$\mathsf{EK}_{\mathcal{R}}$	Attack
E_4	No	x	-	✓	✓	x	KCI
E_8	Yes	x	✓	✓	✓	x	KCI

Table 5.3: The two cases of the eCK/CK+ model which are NOT satisfied by Signal's X3DH, and so are not included in our model. This lack of KCI is exactly where these protocols break down.

Because the semi-static key is reused and not actually ephemeral, we do not assume it is simply a long-term key in the other events of Table 5.2. In the KCI attacks, we allow it to be revealed as both ephemeral and long-term, to properly capture various forms of key-leakage that could lead to that attack and to strengthen the model (as mentioned above).

The MEX cases are more interesting, however. The original Signal X3DH protocol is not secure if the semi-static key can be revealed in cases E_2, E_3, and E_6. Hence, they are set to x in Table 5.2 due to our goal of accurately capturing the security of this original Signal protocol. In the spirit of the MEX property, the protocol would ideally be secure even when these three cases allowed SK to be revealed—there is no reason to treat the semi-static key as long-term in these cases. As we will show later, our new protocol (SI-X3DH) is secure even if these three cases marked by asterisks are changed to ✓.

5.3.2 Further security properties

We briefly discuss (full) perfect forward secrecy (PFS) as opposed to just weak PFS, which is proved in the model above. Krawczyk [Kra05] shows that any two-message key exchange protocol authenticated via public keys (without a secure shared state already established) cannot achieve true perfect forward secrecy. Despite this, it is claimed in [MP16b] that X3DH can be considered to have PFS, assuming that the identities of the users can be trusted via some means outside the protocol. In this specific case, Bob's signature on the semi-static key can be used to verify that the semi-static key does indeed belong to Bob, preventing even an active attacker from tampering with the keys Bob provides to defeat PFS (in particular, the server cannot maliciously provide semi-static keys to Alice while pretending they came from Bob). The same holds for our proposed scheme, but will not be discussed further in this thesis—the situation is identical to the original Signal X3DH.

Another very important property of X3DH, which is not captured by the above security model (or in general by the eCK or CK+ models), is that of **deniability**. Deniability has two flavours: offline and online deniability. A protocol is **offline-deniable** if an adversary can gain no non-repudiable evidence of message authorship from a transcript even if the long-term keys involved are compromised. On the other hand, **online-deniability** means that even by interacting with the target (or colluding with another user with whom the target interacts), the adversary cannot gain any such evidence. A protocol satisfying both offline and online deniability is known as **strongly-deniable**. Unfortunately, the Signal protocol fails to achieve online-deniability, as shown by Unger and Goldberg [UG18]—although this notion is very difficult to obtain and arguably less important than offline-deniability. The first formal proof that offline-deniability is indeed achieved by Signal was given by Vatandas, Gennaro, Ithurburn, and Krawczyk [VGIK20].

The proof of offline-deniability for Signal carries over to our protocol in an essentially identical manner, because of how similar the two protocols are. The proof reduces to the Knowledge of DH (KDH) assumption and its variants (K2DH and EKDH) which informally state that it should be infeasible for an adversary, given as input public keys for which the secret keys are unknown, to output DH values and other public keys they do not know the secret key to, yet still satisfy relationships of the form $\mathsf{dh}_i = \mathrm{DH}(K_1, K_2)$ (where K_1, K_2 are public keys). We will not formally define the assumptions here, but refer the reader to [VGIK20]. We give a brief, informal outline of this proof in Section 5.5.4.

5.4 Using SIDH for post-quantum X3DH

Suppose, first, that we naively drop SIDH in as a replacement for DH in Figure 5.1. In order to prevent adaptive attacks from either party, it suffices to require proof that certain public keys are honestly generated (for example, requiring proof that each member knows their corresponding private key). In the case of EK_A, this could easily be done through an FO-type transformation [HHK17], as was done in SIKE [ACC$^+$17] (discussed in Section 3.4).

However, upon further examination, we notice that Bob's semi-static public key poses an issue. As Bob may be offline at the time of exchange, and this key will be reused across multiple iterations of the protocol, he cannot reveal the secret key to Alice. Even if EK_A is proven to be honestly generated, there would be a concrete attack allowed in the CK security model, despite Galbraith's [Gal18, A.3] claim that using an ephemeral key in the exchange introduces enough randomness to prevent information about the long-term secret being leaked. Precisely, in CK-type models, the adversary can use a reveal query on the private key of EK_A to essentially remove the protection it provides, and then perform an adaptive attack using a malicious semi-static key. The best we can hope for then is that Bob also provides a non-interactive proof of honest generation of SK_B. Unfortunately, because the key SK_B is regularly rotated, such a proof would have to be regenerated and reverified every time, and these proofs are not (currently) efficient enough to make this an attractive course of action.

Instead, we opt to modify the original X3DH protocol somewhat, so that SK_B is not used in a key exchange with IK_A (temporarily removing dh_1 from Figure 5.1, which we shall soon replace). This means that even if Bob maliciously adapts SK_B in order to learn Alice's key, the only key he could learn is the secret to EK_A, which is ephemeral and revealed to him using the FO transformation anyway. The other components, $\mathsf{dh}_2, \mathsf{dh}_3$, and dh_4, all involve only Alice's provably honest ephemeral key, so neither party can learn anything in these exchanges. Therefore, the only thing left to resolve is how to replace dh_1 so that IK_A is still used safely to implicitly authenticate Alice. We cannot use an exchange $\mathrm{SIDH}(\mathsf{IK}_A, \mathsf{EK}_B)$ for a symmetrical reason to above, even if we ignored the fact that EK_B is only optional. Thus, to include the key IK_A in the exchange to authenticate Alice, we are left only with one option: $\mathsf{dh}_1 = \mathrm{SIDH}(\mathsf{IK}_A, \mathsf{IK}_B)$.

In this case, we must prove that the long-term keys $\mathsf{IK}_A, \mathsf{IK}_B$ are honestly generated, to ensure an adaptive attack cannot be performed by registering multiple fake users with adaptive public identity keys. Because these keys are fixed and registered (or even authenticated) in advance, we do not encounter the efficiency degradation of using a more expensive proof to prove knowledge of the corresponding secret keys—a proof would have to be verified only once per new contact. In fact, depending on the trust model we use for the server, the verification of these proofs could be offloaded to the server at registration time and would have no impact on users. If we do not wish to place such trust in the server, it is simple to verify these proofs out-of-band at the time of first communication with any new contact. In fact, the Signal X3DH protocol already assumes that participants will authenticate each other's identity public key via some unspecified external channel, depending on the desired trust model [MP16b]. The Signal Private Messenger app presents "safety numbers" and QR codes that can be used to verify contacts in-person. Thus, the introduction of these proofs does not change the trust model of Signal. Proving SIDH public keys are honestly generated can be done using a non-interactive zero-knowledge (NIZK) Proof of Knowledge (PoK) of the corresponding secret key. In Chapter 4 we present such a proof protocol and show that using it as part of a non-interactive key exchange is much more efficient than resorting to other protocols such as k-SIDH (in terms of isogeny computations) or generic NIZK proof systems. Note that the definition of distributional zero-knowledge that we used in Chapter 4 is sufficient for this application, because the keys in the protocol are generated randomly in accordance with the usual SIDH key generation algorithm Gen. Thus, this SIDH PoK is perfectly suitable for our situation.

Exactly as in Signal's X3DH, we still also require a signature by Bob on SK_B, to ensure that the server does not fake SK_B and break perfect forward secrecy by later corrupting IK_B (one of the adversarial abilities in our security model). This poses another obstruction to efficiency, because using an SIDH signature here would require sending and verifying such a signature regularly—every time Bob replaces his semi-static key. SIDH signatures are inefficient, and we do not recommend their use for practical systems where signatures need to be regularly created and verified. Instead, we suggest using another post-quantum signature scheme, such as a hash-based signature. The ability to use any post-quantum signature scheme for this purpose was already discussed in Section 5.1. Whichever verification key Bob uses for these signatures should be registered (and verified) in advance, just as the identity keys are.

If IK_A and IK_B are proven to be honestly generated then we can use $\mathsf{dh}_1 = \mathrm{SIDH}(\mathsf{IK}_A, \mathsf{IK}_B)$ in the exchange without risk of adaptive attack. Historically, $H(E_{AB}, E_{XY})$ type protocols are referred to as the "unified model". A naive scheme of this form was shown to be vulnerable to interleaving and known-key attacks by Blake-Wilson, Johnson, and Menezes [BJM97, Protocol 3]. Essentially, the adversary starts two sessions with the same user: $\Pi^s_{i,j}$ and $\Pi^u_{i,j}$ (participant i thinking they are communicating with j for the s- and u-th time, respectively). In each of these two sessions, the ephemeral key E_u (or E_s) provided by i is forwarded to the other session, and given back to i (as if coming from j). Then the shared key of both sessions will be $H(E_{ij}, E_{us})$. Revealing either of the two session keys will reveal the session key of the other. For comparison, a protocol of the form $H(E_{AY}, E_{BX})$ has that $H(E_{js}, E_{iu}) \neq H(E_{ju}, E_{is})$, so the attack would not be possible. Including the ephemeral keys E_s and E_u individually in the hash too would prevent this attack, because the ordering would differ between the two sessions. Jeong, Katz, and Lee [JKL04] prove this to be secure ($\mathcal{TS}2$) in the ROM provided knowledge of the secret keys is proven. In the Signal case, because we additionally have $\mathsf{dh}_2 = \mathrm{SIDH}(\mathsf{EK}_A, \mathsf{IK}_B)$ in the exchange, this symmetry between sender and receiver is already broken. Therefore, we claim that our modified dh_1 computation is secure.

One other disadvantage of this modification is that it impacts the KCI resistance of the scheme. That is, if the adversary corrupted IK_B, they could pretend to be Alice by choosing any ephemeral key they like, and calculating dh_1 using the known secret key, so Bob would accept it as coming from Alice herself. However, as above, this was the case with the original Signal X3DH anyway (if SK_B was corrupted). It is important to note that due to this modification, the impersonation can persist for longer than in X3DH, since corruption is no longer repaired by the regular replacement of SK_B. While worthy of consideration, we believe the change is acceptable. As mentioned in the introduction of this chapter, medium-term impersonation seems just as damaging as long-term, and corruption of an identity key is a severe break in security anyway. Because neither scheme can claim to have KCI resistance, we still assert that SI-X3DH satisfies the same security requirements as Signal X3DH, despite this practical difference.

Unlike traditional Diffie–Hellman, where both participants' keys are of the form g^x, in SIDH we have an asymmetric setup—one user uses a degree-$\ell_1^{e_1}$ isogeny, while the other uses a degree-$\ell_2^{e_2}$ isogeny. In order to make this work in X3DH where users can be both initiators and receivers, we require that each user has two long-term identity keys: one of each degree. For concreteness, we shall assume that $\ell_1 = 2$ and $\ell_2 = 3$, therefore the isogenies used by Alice and Bob have degree 2^{e_1} and degree 3^{e_2} respectively. The 3^{e_2}-isogeny key is used when initiating a key exchange (that is, by Alice), and the 2^{e_1}-isogeny key is used by the receiver (Bob), so that there is no ambiguity or incompatibility. This arrangement is chosen so that the sender has a slightly higher computational burden than the receiver.

All the semi-static keys Bob uploads to the third-party keyserver should thus be generated from 2^{e_1}-isogenies, as should his one-time (ephemeral) keys be. Whenever Alice initiates a key exchange, her ephemeral key should be a 3^{e_2}-isogeny key. Then all three (or four) SIDH exchanges used in the protocol will work as usual.

Thus, we arrive at our modified protocol, which we call **SI-X3DH** (Supersingular Isogeny X3DH). The protocol is given in Figure 5.2. In each instance of the protocol, Alice requests Bob's public key package from the server, as before. This key package includes Bob's signature verification key VK_B, which is used to validate the signature on his semi-static key SK_B. Alice will then generate a random seed s and use a preimage resistant hash function H_1 to compute an ephemeral secret key $\mathsf{sk}_e \leftarrow H_1(s)$. Let the corresponding public key be denoted EK_A. She will then compute the pre-shared key PSK, and an FO-proof π as follows:

$$\begin{aligned}
&\mathsf{dh}_1 = \mathrm{SIDH}(\mathsf{IK}_A, \mathsf{IK}_B),\\
&\mathsf{dh}_2 = \mathrm{SIDH}(\mathsf{EK}_A, \mathsf{IK}_B),\\
&\mathsf{dh}_3 = \mathrm{SIDH}(\mathsf{EK}_A, \mathsf{SK}_B),\\
&\boxed{\mathsf{dh}_4 = \mathrm{SIDH}(\mathsf{EK}_A, \mathsf{EK}_B)},\\
&\mathsf{PSK} = \mathrm{KDF}(\mathsf{dh}_1 \parallel \mathsf{dh}_2 \parallel \mathsf{dh}_3 \boxed{\parallel \mathsf{dh}_4}\,),\\
&\pi = s \oplus H_2(\mathsf{dh}_1) \oplus H_2(\mathsf{dh}_2) \oplus H_2(\mathsf{dh}_3) \boxed{\oplus H_2(\mathsf{dh}_4)}\,.
\end{aligned} \tag{5.1}$$

H_1 and H_2 are the same PRGs used in Section 2.3.2. The reason π takes this form will be clear from the security proof we present in Section 5.5.

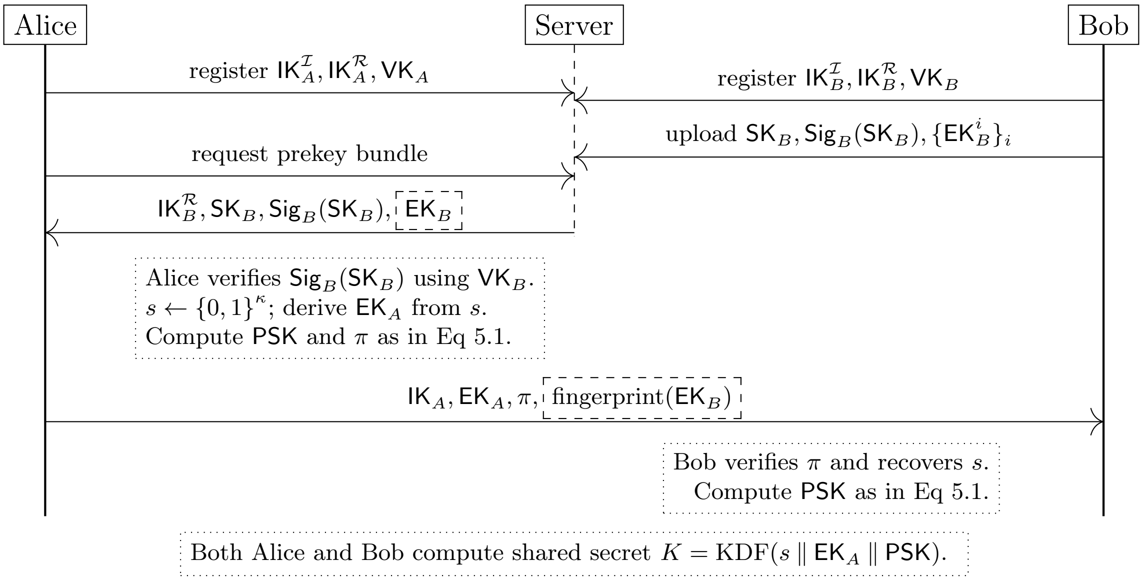

Alice
Server
Bob
register $\mathsf{IK}_A^{\mathcal{I}}, \mathsf{IK}_A^{\mathcal{R}}, \mathsf{VK}_A$
register $\mathsf{IK}_B^{\mathcal{I}}, \mathsf{IK}_B^{\mathcal{R}}, \mathsf{VK}_B$
upload $\mathsf{SK}_B, \mathsf{Sig}_B(\mathsf{SK}_B), \{\mathsf{EK}_B^i\}_i$
request prekey bundle
$\mathsf{IK}_B^{\mathcal{R}}, \mathsf{SK}_B, \mathsf{Sig}_B(\mathsf{SK}_B), \mathsf{EK}_B$
Alice verifies $\mathsf{Sig}_B(\mathsf{SK}_B)$ using VK_B.
$s \leftarrow \{0,1\}^\kappa$; derive EK_A from s.
Compute PSK and π as in Eq 5.1.
$\mathsf{IK}_A, \mathsf{EK}_A, \pi,$ fingerprint(EK_B)
Bob verifies π and recovers s.
Compute PSK as in Eq 5.1.
Both Alice and Bob compute shared secret $K = \mathrm{KDF}(s \parallel \mathsf{EK}_A \parallel \mathsf{PSK})$.

Figure 5.2: The SI-X3DH protocol.

Alice then sends (EK_A, π) to Bob, along with an identifier for herself, and information about which of his ephemeral keys she used in the exchange (if any). Bob can check π is valid and honest by re-computing PSK' using IK_A and EK_A, computing s' from π by XORing with the values $H_2(\mathsf{dh}_j)$ (for $j = 1, 2, 3$, and if used, 4), then recomputing $\mathsf{sk}_e' \leftarrow H_1(s')$, and checking that the corresponding public key is equal to EK_A. He computes PSK as in Equation 5.1. If the verification of π succeeded, both Alice and Bob can compute the shared secret $K = \mathrm{KDF}(s \parallel \mathsf{EK}_A \parallel \mathsf{PSK})$. However, if verification failed, Bob should instead choose a random $r \leftarrow \{0,1\}^\kappa$ and compute $K = \mathrm{KDF}(r \parallel \mathsf{EK}_A \parallel \mathsf{PSK})$. This way, his key will not match the one Alice derives with overwhelming probability, and the exchange fails, with Alice learning no information about the cause of failure (or about Bob's secret keys).

5.5 Proof of security

Theorem 5.4. *The SI-X3DH protocol presented in Section 5.4 is secure (correct and sound) in the Signal-adapted-CK model of Definition 5.2, in the random oracle model (where H_1, H_2 and* KDF *are modelled as random oracles), assuming the SI-CDH problem is hard.*

Proof sketch: We briefly outline the proof methodology. The proof is similar to the one given by Cohn-Gordon et al. [CCD+20], refitted to our Signal-adapted-CK model and using the Verifiable and Honest SI-CDH assumptions instead of the standard DDH oracle in the gap assumption. In particular, the Verifiable SI-CDH problem provides an oracle reminiscent of the usual DDH oracle but with fixed keys, rather than taking keys as input. This important weakening of the oracle allows us to achieve security based on SI-CDH without requiring a gap assumption here (and a similar approach could be taken with the classical X3DH protocol too if desired). Cases E_2, E_3, and E_6 require IK_A and IK_B not to be revealed, so we use that as the basis for security in those cases. Similarly, cases E_1 and E_7 will use the fact that EK_A and IK_B are not revealed, and case E_5 relies on EK_A and SK_B not being revealed. Informally, the proof begins by forming a game in which the challenger guesses in advance which session will be tested, as well as the peer ID of that session. The challenger then simulates the game and inserts a VCDH or HCDH challenge into that predicted session, showing that an adversary winning the game can be used to successfully solve the respective hard problem. Once the cases are combined, this gives a proof of soundness of the SI-X3DH protocol.

Proof. It is clear that two parties following the protocol honestly will become partners. It is also clear that they will both successfully derive the same session key and enter an `accept` state, as an SIDH protocol has no failure probability if both parties are faithful. Thus, the SI-X3DH protocol is *correct.*

To prove soundness, we will use a series of game hops. The proof will require splitting into cases following Table 5.2. Games 0 to 3 are common to all cases; we then break into a case-by-case proof.

Game 0. This game is equal to the security experiment in Section 5.3.1. The advantage of the adversary in this game is Adv_0. All queries to the random oracles (H_1, H_2, KDF) are simulated in an on-the-fly manner, and a table of (query, result) pairs is stored.

Game 1. We ensure all honestly generated SIDH keys are unique, or in other words, that there are no key collisions. If a key is generated that collides with any previously generated key, the challenger aborts and the adversary loses the game. With at most n parties, S sessions per party, m medium-term (semi-static) keys per party, we have at most $n + nm + nS$ receiving (2^{e_1}-isogeny) keys, and at most $n + nS$ sending (3^{e_2}-isogeny) keys. A collision among these keys is an instance of the generalised birthday problem, which we now briefly recall.

If M is the size of the domain from which $N \leq M$ objects are uniformly drawn, the generalised birthday problem shows that the probability of a collision between two objects is

$$p(N; M) = 1 - \prod_{k=1}^{N-1} \left(1 - \frac{k}{M}\right). \tag{5.2}$$

So,

$$\mathsf{Adv}_0 \leq p(n + nm + nS; |\mathcal{K}_2|) + p(n + nS; |\mathcal{K}_3|) + \mathsf{Adv}_1.$$

To be explicit, the size of an ℓ^e-isogeny keyspace is

$$(\ell + 1) \cdot \ell^{e-1}, \tag{5.3}$$

so $|\mathcal{K}_2| = 3 \cdot 2^{e_1 - 1}$ and $|\mathcal{K}_3| = 4 \cdot 3^{e_2 - 1}$. Note that the difference between Adv_0 and Adv_1 is therefore negligible, since the numerator in the collision probability is polynomially-sized while the denominator is exponential.

Game 2. We guess in advance which session Π_u^i the adversary will call the Test query against, and abort if this guess is incorrect. Note that we abort with high probability—there is only a $1/nS$ chance of success—but the advantages still only differ by a polynomial factor.

$$\mathsf{Adv}_1 = nS\mathsf{Adv}_2.$$

Game 3. In this game, we guess in advance the index of the peer of the test session Π_u^i—we guess a $v \in \{1, \ldots, n\}$ and abort if $\Pi_u^i.\mathsf{peer\_id} \neq v$. The probability of guessing v correctly is $1/n$, so

$$\mathsf{Adv}_2 \leq n\mathsf{Adv}_3.$$

We now split into cases based on Table 5.2. The cases will be grouped by the approach we take to reduce each case to the VCDH and HCDH hard problems. Specifically, in each scenario, we consider which of the SIDH exchanges is *not* compromised by reveal queries (that is, which of the edges in the exchange graph is not covered by the revealed vertices), and embed the hard problem into that pair of keys. Firstly, we address the MEX events, where neither IK_A nor IK_B are revealed—cases E_2, E_3, and E_6. We then treat the KCI events, cases E_1 and E_7, where EK_A and IK_B remain unrevealed. Finally, we come to the wPFS event, E_5, in which the adversary does not reveal either EK_A or SK_B.

We shall have, overall, that

$$\mathsf{Adv}_3 = \mathsf{Adv}_3^{2,3,6} + \mathsf{Adv}_3^{1,7} + \mathsf{Adv}_3^{5}.$$

5.5.1 Cases E_2, E_3, E_6 (MEX)

As mentioned above, the three cases E_2, E_3, and E_6 all rely on IK_A and IK_B not being revealed—the adversary should thus be unable to compute $\text{SIDH}(\mathsf{IK}_A, \mathsf{IK}_B)$. This is the basis for the following part of the security proof.

Game 4. In this game, we abort if the adversary queries $\mathsf{dh}_1 = \text{SIDH}(\mathsf{IK}_A, \mathsf{IK}_B)$ as the first component of a call to the KDF oracle. We call this event abort_4.

Whenever abort_4 occurs, we show that we can construct an algorithm $\mathcal{B}$ that can solve the Verifiable SI-CDH problem (VCDH) in Definition 2.17. As per that problem, $\mathcal{B}$ receives a triple $(E_A, E_B, \mathcal{O})$. $\mathcal{B}$ will simulate Game 3, except that it replaces IK_u with E_A and IK_v with E_B. It is guaranteed by freshness that $\mathcal{B}$ will never have to output the corresponding (unknown) secret keys. However, these two keys may be used in other sessions, so $\mathcal{B}$ must be able to behave in a consistent manner even when these keys are involved. Specifically, there are only two cases in which $\mathcal{B}$ is unable to compute the session key:

1. A non-tested session between the same users u, v where u is the initiator and v is the responder.
2. A non-tested session between any user other than u, and v, where v is the responder.

In the first of these two cases, the simulator does not know $\text{SIDH}(E_A, E_B)$, which is needed for two reasons: $\mathcal{B}$ needs it to compute the session key, but it is also the solution to the VCDH challenge. In the second case, the simulator does not know $\text{SIDH}(\mathsf{EK}_E, E_B)$ for potentially malicious ephemeral key EK_E, whose secret key is unknown to $\mathcal{B}$. In all other situations, $\mathcal{B}$ will know at least one of the secret keys involved in each SIDH exchange because they were all generated by the challenger.

We begin with the first case. If a session key or ephemeral key reveal query is made on such a session, $\mathcal{B}$ returns a random key. $\mathcal{B}$ also maintains a list of these random keys it generated, and correspondingly the public keys which *should* have been used to compute each one. Then, to ensure that other KDF queries made are consistent with these replaced keys, we do the following on receipt of a query $\text{KDF}(\mathsf{dh}_1 \parallel \mathsf{dh}_2 \parallel \mathsf{dh}_3)$: $\mathcal{B}$ will query $\mathcal{O}(\mathsf{dh}_1)$, and if 1 is returned, this is exactly the case where abort_4 occurs—then $\mathcal{B}$ can return dh_1 as the answer to the VCDH challenge. Otherwise, $\mathcal{B}$ samples a new random key to return as the KDF response, and updates its list accordingly.

In the second case, we involve the FO-proof π_E also sent as part of the key exchange—a proof of honest generation for EK_E. In such a session, $\mathcal{B}$ will check through the output table of queries $\mathcal{A}$ has made to oracle H_2 (which can only have polynomially-many entries). Let IK_w be the identity key of the initiator. For each pair of entries (h, h'), we check whether $H_1(\pi_E \oplus h \oplus h' \oplus H_2(\mathrm{SIDH}(\mathsf{IK}_w, E_B)))$ is the secret key of EK_E. The simulator can always compute $\mathrm{SIDH}(\mathsf{IK}_w, E_B)$ when $w \neq u$ because it knows the private key for IK_w. In order for π_E to be valid, it must have the form

$$\pi_E = s_E \oplus H_2(\mathrm{SIDH}(\mathsf{IK}_w, E_B)) \oplus H_2(\mathsf{dh}_2) \oplus H_2(\mathsf{dh}_3)$$

so the only way for the adversary to have honestly generated π_E is for it to have queried H_2 on inputs $\mathsf{dh}_2, \mathsf{dh}_3$. Therefore, searching through all pairs (h, h') of queries will always result in recovery of s_E if π_E is valid, and if no such pair exists, the receiver would reject the FO-proof and fail the exchange. If such a pair is found, we can use the computed secret key to also compute $\mathrm{SIDH}(\mathsf{EK}_E, E_B)$. $\mathcal{B}$ can now use this j-invariant in a query to KDF to compute a consistent session key.

Thus, $\mathsf{Adv}(\mathsf{abort}_4) = \mathsf{Adv}^{\mathrm{vcdh}}(\mathcal{B})$ and

$$\mathsf{Adv}_3^{2,3,6} \leq \mathsf{Adv}^{\mathrm{vcdh}}(\mathcal{B}) + \mathsf{Adv}_4.$$

Game 5. In this game, we replace the session key of the test session with a uniformly random key. Because Game 4 aborts whenever a KDF oracle query is made involving dh_1, we know in this game that the adversary never queried KDF to get the true session key. Hence, the advantage of winning this game is

$$\mathsf{Adv}_4 = \mathsf{Adv}_5 = 0.$$

Therefore, we have

$$\mathsf{Adv}_3^{2,3,6} \leq \mathsf{Adv}^{\mathrm{vcdh}}(\mathcal{B}).$$

5.5.2 Cases E_1, E_7

These two cases rely on EK_A and IK_B not being revealed. Then $\mathsf{dh}_2 = \mathrm{SIDH}(\mathsf{EK}_A, \mathsf{IK}_B)$ should be unknown to the adversary. The proof is very similar to the first cases above, but now relies on the Honest SI-CDH assumption from Definition 2.19. The main difference is that now, we must guess which of the signed semi-static keys will be used in the test session.

Game 4′. In this game, the challenger guesses the index $j \in \{1, \ldots, m\}$, such that signed semi-static key SK_v^j is used in the test session, and aborts if this guess is wrong. Consequently,

$$\mathsf{Adv}_3^{1,7} \leq m\mathsf{Adv}_{4'}.$$

Game 5′ and 6′. In Game 5′, we abort if the adversary queries the KDF oracle with second component dh_2, equal to the test session's dh_2 component (derived from EK_u and IK_v). Once again, $\mathcal{B}$ will simulate Game 4′. After receiving an HCDH instance triple (E_A, π, E_B), $\mathcal{B}$ will replace the ephemeral key of the test session with E_A, and IK_v with E_B. $\mathcal{B}$ will then also replace the test session FO-proof with $\pi_T := \pi \oplus H_2(\mathrm{SIDH}(E_A, \mathsf{SK}_v^j)) \oplus H_2(\mathrm{SIDH}(\mathsf{IK}_u, E_B))$. Recall from the definition of the HCDH problem, that π already includes the component $H_2(\text{SIDH}(E_A, E_B))$, as required, so π_T has the correct form.

There are two cases in which $\mathcal{B}$ will not be able to compute valid session keys for non-tested sessions. The first is for a session where any user initiates with $\mathsf{EK}_E \neq \mathsf{EK}_u$, and v is the responder. This is because $\text{SIDH}(\mathsf{EK}_E, E_B)$ is unknown when the secret key of EK_E is unknown. The second case is a special case of the first, when EK_u is reused in an exchange with v as the responder. As above, at least one secret key is known in all other situations, so these are the only two SIDH exchanges unable to be computed by $\mathcal{B}$.

In the first case, $\mathcal{B}$ will look up all pairs (h, h') in the polynomial-length output table of queries $\mathcal{A}$ has made to H_2. Suppose IK_w is the identity key of the initiator, and π_E is the FO-proof sent along with the ephemeral key EK_E. $\mathcal{B}$ will check whether $H_1(\pi_E \oplus h \oplus h' \oplus H_2(\text{SIDH}(\mathsf{IK}_w, E_B)))$ is the secret key of EK_E. As above, $\text{SIDH}(\mathsf{IK}_w, E_B)$ is known to $\mathcal{B}$ since the secret key of IK_w is. Also as above, the only way for the adversary to have generated a valid proof π_E is if they had made queries $H_2(\mathsf{dh}_2)$ and $H_2(\mathsf{dh}_3)$—otherwise, even if the adversary guessed the outputs of H_2 correctly (with negligible probability), they would not be able to verify that the π_E they created was actually correct without making the required queries to H_2 anyway. Hence, the only case the proof π_E is accepted is when a valid pair (h, h') exists in the query list of H_2, and if such a pair is found, we can use the secret key to compute the needed j-invariant $\text{SIDH}(\mathsf{EK}_E, E_B)$. $\mathcal{B}$ can now use this j-invariant in a query to KDF to compute a consistent session key. If no pair is found, the receiver would reject the FO-proof and fail the exchange.

In the second case, we cannot compute the output of KDF because $\mathsf{dh}_2 = \text{SIDH}(E_A, E_B)$ is unknown. So $\mathcal{B}$ will return a random key and keep a table for consistency as in the previous cases. Whenever the adversary makes a query to the KDF oracle, we check if $H_1(\pi \oplus H_2(\mathsf{dh}_2))$ corresponds to the secret key of E_A, and if it does, $\mathcal{B}$ has learned dh_2 as the SI-CDH value of E_A and E_B, this is also the case in which the game aborts. Note that the π used here is the one from the HCDH challenge, not from the exchange (π_E) or the test session (π_T). There is a negligible probability $1/2^\kappa$ that the adversary guessed the correct output of H_2 without making a query of the form $H_2(\mathsf{dh}_2)$ (leading to an abort without recovering the answer to the HCDH challenge).

Game $6'$ is identical to Game 5 in the previous section. We therefore have

$$\mathsf{Adv}_3^{1,7} \leq m(\mathsf{Adv}^{\text{hcdh}}(\mathcal{B}) + 1/2^\kappa).$$

5.5.3 Case E_5 (wPFS)

This case relies on EK_A and SK_B not being revealed (wPFS assumes that, in the future, these secrets are unrecoverable). Alternatively, this proof could be reduced to EK_A and EK_B which are both purely ephemeral. However, because EK_B is optional in the Signal protocol (to avoid key exhaustion DoS), we reduce to the former scenario. In this case, we must again guess which of the signed semi-static keys will be used in the test session.

Game $4''$. In this game, the challenger guesses the index $j \in \{1, \ldots, m\}$, such that signed semi-static key SK_v^j is used in the test session. The game aborts if this guess is wrong. Hence,

$$\mathsf{Adv}_3^5 \leq n_m \mathsf{Adv}_{4''}.$$

Game $5''$ and $6''$. These proceed exactly as in Games $5'$ and $6'$ of cases E_1 and E_7 above, but with the HCDH challenge keys inserted into EK_u and SK_v^j. Furthermore, exactly as in the previous subsections, $\mathcal{B}$ knows the secret keys needed to compute the SIDH values of all exchanges except in two cases: an exchange with v as the responder using semi-static key SK_v^j (because EK_E is unknown and potentially maliciously chosen), and the specific subcase where $\mathsf{EK}_E = \mathsf{EK}_u$. This is essentially identical to cases E_1 and E_7. We conclude that

$$\mathsf{Adv}_3^5 \leq m(\mathsf{Adv}^{\text{hcdh}}(\mathcal{B}) + 1/2^\kappa).$$

Finally, bringing all the game hops and cases together, we have

$$\begin{aligned} \mathsf{Adv}_{n,m,S}^{\text{kie}} \;\leq\; & p(n + nm + nS; |\mathcal{K}_2|) \\ & + p(n + nS; |\mathcal{K}_3|) \\ & + n^2 S[\mathsf{Adv}^{\text{vcdh}} + 2m\mathsf{Adv}^{\text{hcdh}} + m/2^{\kappa-1}], \end{aligned} \tag{5.4}$$

where n is the number of participants, m is the number of semi-static keys per participant, and S is the maximum number of sessions run per party.

Because the VCDH and HCDH problems are hard if the SI-CDH problem is (shown in Section 2.3.2), we have that SI-X3DH is secure if the SI-CDH problem is hard, as required. □

5.5.4 Deniability

As mentioned in Section 5.3.2, the proof of offline-deniability of SI-X3DH is almost identical to that of the original Signal X3DH protocol (given in [VGIK20]), due to the similarity between the schemes. We give a brief informal outline of the proof below.

Proof outline: Intuitively, for Bob to prove Alice's involvement, he would have to provide a Diffie–Hellman value $\mathrm{DH}(A, \cdot)$ which he could not possibly have generated himself—it must therefore have been generated by Alice. Because no DH values are exchanged between Alice and Bob in X3DH or SI-X3DH, and because the KDH, K2DH and/or EKDH assumptions hold, this is impossible. On top of this, because neither protocol uses a signature on session-specific information (unlike [HKKP21]), there is no loss of deniability there either. Proof of offline-deniability proceeds as an argument about simulatability, which we shall now sketch.

In the case of deniability for the initiator, given Alice's public key IK_A, the simulator Sim will generate $x \leftarrow \mathcal{K}_3$ and compute EK_A. Sim will then send this to Bob, who outputs keys IK_B, $\mathsf{SK}_B, \mathsf{EK}_B$. The simulator can compute $\mathsf{dh}_2 = \mathrm{SIDH}(\mathsf{EK}_A, \mathsf{IK}_B)$, $\mathsf{dh}_3 = \mathrm{SIDH}(\mathsf{EK}_A, \mathsf{SK}_B)$, and $\mathsf{dh}_4 = \mathrm{SIDH}(\mathsf{EK}_A, \mathsf{EK}_B)$ because x is known, but cannot compute $\mathrm{SIDH}(\mathsf{IK}_A, \mathsf{IK}_B)$. Under the KDH-type assumptions, there must be an extractor $\hat{\mathcal{B}}$ for Bob's key IK_B—let us call it $\hat{\mathcal{B}}$. If $\hat{\mathcal{B}}$ outputs $\hat{Z}$ then the shared key is $\mathrm{KDF}(\hat{Z} \parallel \mathsf{dh}_2 \parallel \mathsf{dh}_3 \parallel \mathsf{dh}_4)$—the real shared key. On the other hand, if $\hat{\mathcal{B}}$ outputs $\perp$, then Sim chooses a session key at random. In either case, Sim also computes the FO-proof π using the session key it computed. In the second case, no PPT algorithm can compute $\mathrm{SIDH}(\mathsf{IK}_A, \mathsf{IK}_B)$ without knowing IK_B, so the random key is indistinguishable from the real key.

We come now to the case of deniability for the responder, given Bob's public key IK_B, and also a signed semi-static key $\mathsf{SK}_B, \mathsf{Sig}_B(\mathsf{SK}_B)$. The simulator will send these two public keys to Alice, who outputs a key EK_A. Under the KDH-type assumptions, there exists an extractor $\hat{\mathcal{A}}$ for Alice which will either output the required SIDH values needed to compute the real key or will fail to output, in which case a random key will be indistinguishable from the real one as above. Thus, either way, assuming the KDH, K2DH and EKDH assumptions hold in the SIDH setting (which we claim they do), our SI-X3DH protocol is offline-deniable.

5.6 Efficiency

SIDH is a practically efficient post-quantum key exchange proposal. SIKE, derived from SIDH, is an alternate candidate in round 3 of NIST's post-quantum standardization competition. Duits [Dui19] examined the practical efficiency of using SIDH in the Signal protocol (though note that the implementation is not SI-X3DH, but the naive implementation, vulnerable to adaptive attacks), and found it entirely practical.

The SI-X3DH protocol uses three or four SIDH exchanges as part of the process to derive the shared key—a reflection of how Signal X3DH also uses three or four DH exchanges. In a single SI-X3DH exchange, the only other information sent (on top of the SIDH public keys) is the FO-proof π. This is simply κ bits, which does not have a significant impact on the efficiency of the protocol. Thus, using SIDH for a post-quantum X3DH replacement is efficient at exchange time.

One of the main drawbacks of the SI-X3DH protocol is that it requires registering two keys rather than one on the server—a receiving key and a sending key. This is due to the inherent asymmetry of the SIDH protocol. However, SIDH has among the shortest key sizes of any post-quantum key exchange scheme, so this is not an issue. Note, too, that to initiate a conversation with a peer, only one key is required to be retrieved (the peer's sending key is not needed if they are the responder).

The second major drawback is that these keys also require an SIDH Proof of Knowledge or proof of honest generation, such as the one presented in Chapter 4. Depending on the trust model, this can be offloaded to the server at registration time or verified out-of-band, and only needs to be verified once. The best case is that a user verifies the proof for a contact once and then continues creating sessions with that same contact over a long period of time. However, if users regularly add new contacts, this could create a large overhead by requiring verification of such a proof for each. In the worse case, if a proof is required on nearly every new key exchange session, the overhead would be very large, and our scheme would no longer be efficient.

As discussed earlier, it appears that any post-quantum Signal X3DH replacement requires a post-quantum signature scheme to achieve perfect forward secrecy, and our scheme is no different. However, we emphasise that the use of a single signature is much more efficient than the generic schemes by Hashimoto et al. [HKKP21] and Brendel et al. [BFG$^+$22], which both require two signatures per exchange—one of which must be a more expensive ring or DVS signature to attain deniability.

We now consider the exchange-time efficiency of our protocol compared to the others proposed in the literature. By exchange-time, we mean the protocol occurring *after* the identity keys of the peer have been retrieved and verified (thus not taking into account the SIDH PoK on the identity keys). We consider the exchange-time efficiency because we assume a scenario in which we are beginning a new exchange with an already-verified peer, or a peer whose keys were verified in-person some time in advance.

As mentioned previously, our protocol is more efficient in terms of computation at exchange-time than Brendel et al.'s Split-KEM based X3DH [BFG$^+$20] protocol using CSIDH (assuming CSIDH does even satisfy the security properties needed for their split-KEM scheme, which they leave as an open problem). Based on NIST security level 1, we compare the fast, constant-time CTIDH [BBC$^+$21] implementation of CSIDH-512 with the SIKEp434 parameter set. According to Banegas et al. [BBC$^+$21], the cost of computing the CSIDH action is approximately 125 million Skylake clock cycles, while Cervantes et al. [COR21] state that SIKEp434 key generation and agreement takes around 5 million Skylake clock cycles—roughly 25 times faster. The split-KEM protocol proposed by Brendel et al. would require two CSIDH actions for each of the four encapsulations and decapsulations. SI-X3DH, on the other hand, requires only four SIDH exchanges, so in total would be around 50 times faster.

While the Signal-conforming AKE scheme proposed by Hashimoto et al. [HKKP21] and the SPQR protocol by Brendel et al. [BFG$^+$22] can be instantiated using efficient KEMs such as SIKE or other NIST post-quantum KEM candidates, the need for a post-quantum secure ring signature or DVS scheme is a large drawback to the efficiency of these protocols. Instantiating with the ring signature schemes of Beullens, Katsumata, and Pintore [BKP20], and choosing the lattice-based instantiation (Falafl) to optimise for speed (rather than signature and key size), would require around 78 million clock cycles for signing. Therefore, the signing time alone is already four times slower than the full SI-X3DH key exchange, and such a signature would be around 30 KB in size. The smaller isogeny-based instantiation (Calamari), whose signatures are around 3.6 KB, would take on the order of 10^{11} clock cycles—many orders of magnitude slower.

Thus, concretely, when performing an exchange with a user whose identity key has been verified via an SIDH Proof of Knowledge in advance or out-of-band, SI-X3DH is the fastest exchange-time post-quantum alternative to Signal's X3DH protocol currently in the literature.

Finally, to summarise the key differences with the original Signal X3DH protocol in a short form:

- Users must register two long-term public keys rather than one—a receiving and a sending key.
- Key compromise impersonation attacks (KCI) can no longer be rectified by replacing the semi-static key. Bob needs to switch to a new long-term key if his long-term key is compromised.
- Long-term key registration requires a proof of honest generation (such as the SIDH Proof of Knowledge in Chapter 4), to avoid adaptive attacks by registering many fake users with malicious long-term keys.
- The signatures on Bob's semi-static keys can use any post-quantum signature scheme, and Bob should additionally register his signature verification public key so these can be validated.
- When initiating a new key exchange, Alice must also send a small FO-proof (κ bits in size) along with her ephemeral public key, and Bob must check this proof on its receipt.

Chapter 6

Hyperelliptic Curves and Ideal Class Groups

In Chapter 7, we will discuss the use of class groups and hyperelliptic curves in constructing "groups of unknown order" for zero-knowledge proofs and other applications. This chapter will serve as background, providing some mathematical detail we will rely upon in the next chapter.

Recall from Chapter 1 that an elliptic curve is a non-singular projective algebraic curve of genus one. We can certainly generalise this definition to higher genus, and a special case of these higher-genus curves are the **hyperelliptic curves**—a particular class of smooth, projective algebraic curves of genus greater than one. Note that as curves, these still have *dimension* one—otherwise we would be talking about algebraic surfaces, or more generally, algebraic varieties. As we will see, curves of higher genus no longer have a group structure on their set of points like elliptic curves do. That is because, unlike elliptic curves, higher genus curves are no longer isomorphic to their corresponding Jacobian varieties and thus do not inherit an abelian group structure through this isomorphism in the same way that elliptic curves do. However, we can obtain a group structure on these curves in a different form, also inherited from their Jacobians. This will be explored in Section 6.4.

First, though, we will venture into the realm of ideal class groups of imaginary quadratic fields, and their correspondence with class groups of binary quadratic forms. While ideal class groups were suggested for use in cryptography by Buchmann and Williams in 1988 [BW88], they did not find popularity until much later, when they became useful as groups of unknown order. We will look further at this application in Chapter 7, but will examine some preliminary theory now.

A major result of this chapter is a more compact representation of class group elements. Inspired by a signature compression method of Bleichenbacher [Ble04], in Section 6.3 we compress elements of class groups to 3/4 of the size of their usual representation.

6.1 Ideal class groups

Detailed references for this section include the books of Cohen [Coh10] and of Cox [Cox89].

An **imaginary quadratic field** is an algebraic extension

$$K = \mathbb{Q}(\sqrt{d}) = \left\{ a + b\sqrt{d} \;\middle|\; a, b \in \mathbb{Q} \right\} \tag{6.1}$$

where $d < 0$ is a square-free integer and $\sqrt{d}$ denotes the positive imaginary square root of d. In particular, $\sqrt{d}$ has degree two (i.e., $\sqrt{d}$ satisfies a quadratic form $f(\sqrt{d}) = 0$), so K has dimension two as a vector space over $\mathbb{Q}$. There is a one-to-one correspondence between square-free integers d and quadratic fields K (note that 0 and 1 are not considered square-free). It is the condition that $d < 0$ that makes the field imaginary—those fields where d is positive are called **real**.

Recall that the **ring of integers** of a field K, denoted by $\mathcal{O}_K$, is made up of the **algebraic integers** of the field (the elements which are roots of monic polynomials with integer coefficients). $\mathcal{O}_K$ is a free $\mathbb{Z}$-module, and when K is a quadratic number field, $\mathcal{O}_K$ has dimension two. Consequently, we can specify an integral basis of $\mathcal{O}_{\mathbb{Q}(\sqrt{d})}$ consisting of two elements $\{1, \omega\}$, where

$$\omega = \begin{cases} \frac{1}{2}(1+\sqrt{d}) & \text{when } d \equiv 1 \pmod 4, \\ \sqrt{d} & \text{otherwise.} \end{cases} \tag{6.2}$$

Because $\mathcal{O}_K$ is a subring of K and a full $\mathbb{Z}$-lattice (a finitely generated $\mathbb{Z}$-submodule of K such that $\mathbb{Q} \otimes_{\mathbb{Z}} \mathcal{O}_K = K$) of rank $2 = [K : \mathbb{Q}]$, then $\mathcal{O}_K$ is an **order** in K. In fact, the ring of integers is the unique maximal order of the field and is always a Dedekind domain. We also observe that the **discriminant** Δ of $\mathbb{Q}(\sqrt{d})$, which is defined using the basis $\{1, \omega\}$, is

$$\Delta = \begin{cases} d & \text{when } d \equiv 1 \pmod 4, \\ 4d & \text{otherwise.} \end{cases} \tag{6.3}$$

A discriminant such as this is more precisely known as a **fundamental** discriminant. It is therefore often more convenient to choose a square-free discriminant $\Delta \equiv 1 \pmod 4$, rather than checking that $\Delta/4$ is square-free.

A **fractional ideal** of $\mathcal{O}_K$ is an $\mathcal{O}_K$-submodule J of K, such that $rJ \subseteq \mathcal{O}_K$ for some $r \in \mathcal{O}_K$. Intuitively, r can be thought of as the common denominator of J, and clearing the denominators gives an ideal in the usual sense (also known as an integral ideal, for distinction). Because $\mathcal{O}_K$ is a Dedekind domain, every non-zero fractional ideal J has an inverse

$$J^{-1} = \{x \in K \, : \, xJ \subseteq \mathcal{O}_K\}. \tag{6.4}$$

These non-zero fractional ideals of $\mathcal{O}_K$ form an abelian group, which we denote by J_K. In this group, $(1) = \mathcal{O}_K$ itself is the identity. The product of two fractional ideals I, J is given by

$$IJ = \left\{ \sum a_i b_i \;\middle|\; a_i \in I, b_i \in J \right\}. \tag{6.5}$$

As usual, a **principal** fractional ideal is a fractional ideal generated by a single non-zero element. Let $P_K < J_K$ be the subgroup of principal fractional ideals. Then the **ideal class group** is the quotient group

$$Cl(\mathcal{O}_K) = J_K/P_K. \tag{6.6}$$

In other words, this is the abelian group of fractional ideal classes under the equivalence relation $\mathfrak{a} \sim \mathfrak{b}$ if and only if $(\alpha)\mathfrak{a} = (\beta)\mathfrak{b}$ for some principal ideals $(\alpha), (\beta)$. We denote the class of an ideal $\mathfrak{a}$ as $[\mathfrak{a}]$. The identity of this group is, therefore, $[(1)]$.

The order of $Cl(\mathcal{O}_K)$ is the **class number** of K, denoted by $h(\Delta)$. It is always finite when constructed with a ring of integers as we have described. It follows from the Brauer–Siegel theorem (see [HM00]) that for sufficiently large negative discriminants, on average, the class number satisfies

$$\log h(\Delta) \sim \log \sqrt{|\Delta|} \quad \text{as} \quad \Delta \to -\infty. \tag{6.7}$$

We can therefore conservatively assume $\approx \frac{1}{2} \log_2 |\Delta|$-bit group sizes for cryptographic-sized negative discriminants.

6.2 Form class groups

In practice, it is more efficient to represent and compute in $Cl(\mathcal{O}_K)$ using binary quadratic forms. The ideal class group of an imaginary quadratic field of discriminant $\Delta < 0$ is isomorphic to the **form class group** of the same discriminant Δ. We let (a, b, c) denote the binary quadratic form

$$(a, b, c) = ax^2 + bxy + cy^2 \in \mathbb{Z}[x, y] \quad \text{with} \quad b^2 - 4ac = \Delta\,. \tag{6.8}$$

When Δ is fixed, we can represent this form using only two elements (a, b), because c is uniquely determined by the equation $c = (b^2 - \Delta)/4a$. A form is **primitive** if the greatest common divisor of a, b, and c is 1. This will always be the case for us, since forms whose discriminants Δ are fundamental are always primitive, and we always work with such Δ. A form is **positive definite** if $a > 0$. As with ideal class groups, there is an equivalence relation on these quadratic forms. Two forms f, g are equivalent, $f \sim g$, if

$$f(x, y) = g(\alpha x + \beta y, \gamma x + \delta y) \tag{6.9}$$

for some α, β, γ, and δ in $\mathbb{Z}$ with $\alpha\delta - \beta\gamma = 1$ (that is, if they are in the same orbit under $\mathrm{SL}_2(\mathbb{Z})$). Equivalent forms always have the same discriminant so the class group of forms under this relation, denoted by $Cl(\Delta)$, is well-defined.

We represent each equivalence class in $Cl(\Delta)$ using the unique **reduced form** in that class. A form (a, b, c) is reduced if $|b| \leq a \leq c$, and if $|b| = a$ or $a = c$, then $b \geq 0$. Lagrange [Lag75], and later Gauss [Gau66] and then Zagier [Zag81], gave algorithms to find the equivalent reduced form for any binary quadratic form.

The group law in the form class group is known as composition of forms, and is due to Gauss (it corresponds exactly to the multiplication of ideals in $Cl(\mathcal{O}_K)$ that we defined in the previous section). Composition of forms is well-defined but does not usually output a reduced form, so reduction must be performed as an additional step. We shall not give these algorithms here, but refer the reader to Cohen's book [Coh10]. The identity in $Cl(\Delta)$ is the equivalence class of the principal form: $(1, 0, -k)$ when $\Delta = 4k$, or $(1, 1, k)$ when $\Delta = 4k + 1$.

6.3 Compressing class group elements

We now propose a new method of compressing ideal and form class group elements, based on Bleichenbacher's Rabin signature compression algorithm [Ble04]. In Chapter 7, we will encounter class groups of large discriminant, so being able to compress their elements will be a welcome relief in some situations. We first review the original signature compression algorithm, and then in Section 6.3.2 we show how to apply it to elements of the form class group described above.

The class group element compression algorithm presented in this section is a corrected version of the algorithm in the published version of this work [DGS21].

6.3.1 Bleichenbacher's Rabin signature compression algorithm

A Rabin signature on a message m under the public key N (an RSA modulus) is an integer σ such that

$$\sigma^2 \equiv m \pmod{N}. \tag{6.10}$$

Normally σ is the same size as N, but Bleichenbacher showed how to bring this down to $\sqrt{N}$. The continued fraction algorithm (or the Euclidean algorithm) can be used to compute integers s and t with $0 \leq s < \sqrt{N}$ and $0 < |t| \leq \sqrt{N}$ such that $\sigma t \equiv s \pmod{N}$: see Algorithm 6.1 and Lemma 6.1 below. The compressed signature is t. Given the message m and this compressed signature t, let $x = mt^2 \bmod N$. Then $s^2 \equiv \sigma^2 t^2 \equiv x \pmod{N}$, but $0 \leq s < \sqrt{N}$, so s^2 will always be less than N. Therefore, we can recover s from m and t by taking the *integer* square root of x, ignoring the modulus completely. It is then trivial to recover $\sigma \equiv t^{-1}s \pmod{N}$. Note that if t is not invertible modulo N, then we have found a factor of N, and the signature scheme is broken. Therefore σ is recoverable from t, which is only half the bit-length of σ, so this compression provides a nice saving.

The following algorithm (Algorithm 6.1), presents the partial extended Euclidean algorithm used to find s and t above (on input N, σ). This will be used for our new class group element compression technique too, as we shall see.

Algorithm 6.1 `PartialXGCD`.

```
Input: Integers $a > b > 0$
Output: Integers $s \in [0, \sqrt{a})$ and $t \in [-\sqrt{a}, \sqrt{a}]$ such that $s \equiv bt \pmod{a}$
$(s, s', t, t', u, u') \leftarrow (b, a, 1, 0, 0, 1)$
while $s \geq \sqrt{a}$ do        ▷ Invariants: $0 \leq s < s'$, $s = au + bt$, $s' = au' + bt'$
    $q \leftarrow s' \operatorname{div} s$        ▷ Euclidean division without remainder
    $(s, s', t, t', u, u') \leftarrow (s' - qs, s, t' - qt, t, u' - qu, u)$
return $(s, t)$
```

Lemma 6.1. *Given integers $a > b > 0$, Algorithm 6.1 returns (s, t), such that $s \equiv bt \pmod{a}$, $0 \leq s < \sqrt{a}$, and $0 < |t| \leq \sqrt{a}$.*

Proof. Algorithm 6.1 is a truncated version of the extended Euclidean algorithm, stopping when $s < \sqrt{a}$ rather than $s = 0$. The invariants $s' > s \geq 0$, $s = au + bt$, and $s' = au' + bt'$ are easily verified. It directly follows from $s = au + bt$ that $s \equiv bt \pmod{a}$. Another invariant, $|s't| \leq a$, is proven in [Gal12, Lemma 2.3.3]. It is also proven there that $|t'| \leq |t|$, and since t is initialised to 1, $t \neq 0$. Since s takes a sequence of strictly decreasing values, at some point $0 \leq s < \sqrt{a}$ and $s' \geq \sqrt{a}$; this is where the loop terminates. It remains to show that at this point, we also have $|t| \leq \sqrt{a}$: but this follows from the invariant $|s't| \leq a$ and $s' \geq \sqrt{a}$. □

6.3.2 An improved class group element compression algorithm

Suppose we have a reduced form (a, b) in $Cl(\Delta)$, for a fixed $\Delta < 0$. Since the form is reduced, we have $|b| \leq a < \sqrt{|\Delta|}$, so the pair (a, b) can be encoded in approximately $\log_2 |\Delta|$ bits. This is the traditional "compressed" representation of a class group element.

We can do better than this, though. Since $b^2 - 4ac = \Delta$, we have

$$b^2 \equiv \Delta \pmod{a}, \tag{6.11}$$

a relation reminiscent of the Rabin signature verification equation. The situation is not exactly the same—a is not an RSA modulus, and b is in $(-a, a\,]$ rather than $[\,0, a)$—but it is not difficult to adapt signature compression to class group element compression, encoding the coefficient b in half the space.

First, we reduce to the case where $b \geq 0$: we store the sign of b as $\epsilon = 1$ if $b < 0$, and 0 otherwise, and replace b with $|b|$. We will treat the special cases $a = b$ and $b = 0$ later; in the meantime, we may suppose that $0 < b < a$. Using Algorithm 6.1, we compute integers s and t such that $bt \equiv s \pmod{a}$, $0 \leq s < \sqrt{a}$, and $0 < |t| \leq \sqrt{a}$. Then

$$s^2 \equiv b^2 t^2 \equiv \Delta t^2 \pmod{a}. \tag{6.12}$$

Given a and t, we can compute $x = \Delta t^2 \bmod a$, and then $x = s^2$ *as an integer* because $0 \leq s < \sqrt{a}$. Thus, s can be recovered as the exact (positive) integer square root. Now $bt \equiv s \pmod{a}$, and the Bleichenbacher approach suggests compressing b to t and recovering b as $t^{-1}s \pmod{a}$. Since a is not an RSA modulus, though, we may (and often do) have $\gcd(a, t) \neq 1$, and in this case, t cannot be inverted modulo a.

To fix this, we compress (a, b) to $(a', g, t', b_0, \epsilon)$, where

$$\begin{aligned} g &= \gcd(a, t), \\ a' &= a/g, \\ t' &= t/g, \\ b_0 &= |b| \bmod f. \end{aligned} \tag{6.13}$$

Here, $f \geq g$ is the smallest integer such that $\operatorname{lcm}(f, a') \geq a$, and ϵ and t are defined as above. The reason for this choice of f is that since b is an integer satisfying $0 < b < a$, it is not necessary to compute b by computing $b \pmod{a}$. Instead, we can recover $b \pmod{N}$ for any integer $N \geq a$. Here we use $N = \operatorname{lcm}(f, a')$, where we ensure $N \geq a$, and this f is deterministically computable by the decompression algorithm. This avoids a failure to recover b uniquely, in the case that a' and g share common factors, if we had simply used $f = g$.

To decompress, we compute

$$\begin{aligned} a &= a'g, \\ t &= t'g, \\ x &= t^2\Delta \bmod a, \\ s &= \sqrt{x}. \end{aligned} \tag{6.14}$$

Let $b' \equiv s'(t')^{-1} \pmod{a'}$, where $s' = s/g$. We note that s is always divisible by g since both a and t are, via the invariant $s = au + bt$ from Algorithm 6.1. Then, $b' \equiv b \pmod{a'}$, and we can compute b uniquely from $b \equiv b_0 \pmod{f}$ and $b \equiv b' \pmod{a'}$ using the Chinese Remainder Theorem, since $b < \operatorname{lcm}(f, a')$. Roughly, because two random numbers are coprime with probability $6/\pi^2$ [CJ88], then we expect f to only be a small additive factor higher than g, and therefore that $\log_2 f \approx \log_2 g$. Concretely, we compressed more than 30 million random class group elements of 3845-bit discriminants using an implementation in python, and found that $f - g$ had an average value of 0.1234, and a maximum value of 16, after all these attempts. These numbers did not appear to grow with the size of Δ. Finally, if $\epsilon = 1$ then we correct the sign, replacing b with $-b$. We therefore arrive at the uncompressed form (a, b). If required, c can also be computed as usual with $(b^2 - \Delta)/(4a)$.

For the special case $a = b$, we exceptionally let $t = 0$. This is not ambiguous, because $t = 0$ cannot occur in any other case. The compressed form is then $(a', g, t', b_0, \epsilon) = (1, a, 0, 0, 0)$. For $b = 0$, we compress to $(a, 0, 0, 0, 0)$. Again, this is unambiguous—no other element of $Cl(\Delta)$ compresses to this value.

Algorithms 6.2 and 6.3 make the compression and decompression procedures completely explicit. Note that

$$\log_2 a' + \log_2 g = \log_2 a \approx \log_2 \sqrt{|\Delta|} \tag{6.15}$$

and

$$\log_2 t' + \log_2 b_0 \leq \log_2 t' + \log_2 f \approx \log_2 t \approx \frac{1}{2} \log_2 \sqrt{|\Delta|}\,. \tag{6.16}$$

Algorithm 6.2, therefore, compresses the form (a, b, c) to a $\frac{3}{4}\log_2|\Delta|$-bit representation—three-quarters of the size of the traditional (a, b). When a party receives a compressed group element, it is necessary for them to execute Algorithm 6.3 before performing group operations on the element. Thus, despite the decrease in size, there is some additional overhead when computing with elements in this compressed form.

Algorithm 6.2 Element compression for $Cl(\Delta)$.

Input: A reduced form (a, b, c) in $Cl(\Delta)$, where c may be omitted
Output: A compressed form $(a', g, t', b_0, \epsilon)$

```
if $b = 0$ then
    return $(a, 0, 0, 0, 0)$
if $a = b$ then
    return $(1, a, 0, 0, 0)$
$\epsilon := \begin{cases} 1 & \text{if } b < 0 \\ 0 & \text{otherwise} \end{cases}$
$b := |b|$
$(s, t) \leftarrow$ PartialXGCD$(a, b)$        ▷ Now $s \equiv bt \pmod{a}$, with $0 \leq s < \sqrt{a}$
$g := \gcd(a, t)$
$a' := a/g$
$t' := t/g$
$f := |g|$
while $\mathrm{lcm}(f, a') < a$ do
    $f \leftarrow f + 1$
$b_0 := b \bmod f$
return $(a', g, t', b_0, \epsilon)$
```

Example 6.2. Suppose we have a reduced form $(4, 2, 19)$, where $\Delta = -300$. The partial extended GCD (Algorithm 6.1) of $a = 4$ and $b = 2$ gives us $s = 0$ and $t = -2$, so that

$$0 \equiv -2b \pmod{4}.$$

The GCD of $a = 4$ and $t = -2$ is $g = 2$, so we obtain $a' = 2$ and $t' = -1$. Finally, to compute b_0, we use $f = 3 = g + 1$. This is the smallest integer $f \geq g$ such that $\mathrm{lcm}(f, a') \geq a$, because $\mathrm{lcm}(g, a') = 2 < a$. Then $b_0 = b \bmod 3 = 2$. This gives a final compressed element $(a' = 2, g = 2, t' = -1, b_0 = 2, \epsilon = 0)$.

Algorithm 6.3 Element decompression for $Cl(\Delta)$.

```
Input: A compressed form $(a', g, t', b_0, \epsilon)$ and $\Delta$
Output: A reduced form $(a, b, c)$ in $Cl(\Delta)$, where $c$ may be omitted
if $(g, t', b_0, \epsilon) = (0, 0, 0, 0)$ then
    return $(a', 0, -\Delta/4a')$
if $t' = 0$ then
    Return $(g, g, (g^2 - \Delta)/(4g))$
$a := g \cdot a'$
$t := g \cdot t'$
$x := t^2\Delta \mod a$
$s := \sqrt{x}$                                              ▷ Integer square root
$s' := s/g$                                                  ▷ Exact integer division
$b' := s' \cdot (t')^{-1} \pmod{a'}$
$f := |g|$
while $\mathrm{lcm}(f, a') < a$ do
    $f \leftarrow f + 1$
$b \leftarrow$ ChineseRemainderTheorem$((b', a'), (b_0, f))$   ▷ $b \equiv b' \pmod{a'}$ and $b \equiv b_0 \pmod{f}$
if $\epsilon = 1$ then
    $b := -b$
return $(a, b, (b^2 - \Delta)/4a)$
```

When decompressing, we compute

$$s^2 = t^2\Delta \bmod a = (t'g)^2\Delta \bmod a = -300 \cdot 4 \bmod 2 = 0,$$

and thus recover $s = 0$ successfully as the integer square root. We then compute $b' = s'(t')^{-1} \bmod a' = 0$, giving the congruence $b \equiv 0 \pmod{2}$. Solving the Chinese remainder theorem with $b \equiv 0 \pmod{2}$ and $b \equiv 2 \pmod{3}$ gives the solution $b \equiv 2 \pmod{6}$, and so we successfully deduce that $b = 2$. Note that here, b_0 is equal to b (as an integer), but this will not be true in general.

6.4 Hyperelliptic curves

Good references for more detailed theory of hyperelliptic curves are the books by Menezes, Wu, and Zuccherato [MWZ96] and by Galbraith [Gal12].

Let k be a field of characteristic not 2, with algebraic closure $\overline{k}$. A hyperelliptic curve C of genus g is a curve of the form

$$y^2 + h(x)y = f(x), \tag{6.17}$$

where $h, f \in k[x]$, h has degree at most g, and f is a monic, square-free polynomial of degree $2g + 1$ or $2g + 2$. As in the earlier chapters of this thesis, we will only concern ourselves with fields $k = \mathbb{F}_q$ for a prime power q, which will again be assumed henceforth. For simplicity, we shall further narrow our consideration of hyperelliptic curves to those where $h(x) = 0$ and where $\deg(f) = 2g + 1$ for the remainder of this thesis.

For an extension field K/k, we denote by $C(K)$ the set of points

$$C(K) = \left\{ P \;\middle|\; P \in K^2 \right\} \tag{6.18}$$

satisfying C (the **finite** points over K), together with the projective point at infinity $\infty \in P^2(k)$. Recall that every $P = (x, y)$ on C has an **opposite** point $\tilde{P} = (x, -y)$, with $\infty = \tilde{\infty}$. Unlike points on elliptic curves (which correspond to genus $g = 1$), the points in $C(K)$ do not form a group. Instead, the group we use is the class group of degree-0 divisors on C, also known as the Jacobian. This will be defined below. We require that the affine curve C be non-singular, or in other words, that C has no singular points $(u, v) \in \overline{\mathbb{F}}_q^2$ that satisfy both C and its partial derivatives.

The coordinate ring of a hyperelliptic curve C over K is the quotient ring

$$K[C] = K[x, y] \Big/ (y^2 - f(x)) \ , \tag{6.19}$$

where the modulus is the ideal generated by the equation of C. Elements of $K[C]$ are called polynomial functions on C. Every polynomial function $G(x, y)$ can be written in the form $a(x) - b(x)y$ for some $a, b \in \overline{k}[x]$. The ring $\overline{k}[C]$ is an integral domain.

The function field $K(C)$ of C is the field of fractions of $K[C]$. We say that a function $R \in \overline{k}(C)$ is defined at a point $P \neq \infty$ in $C(\overline{K})$ if and only if there exist polynomial functions $G, H \in \overline{k}[C]$ such that $R = G/H$ and $H(P) \neq 0$. If this holds, then $R(P) = G(P)/H(P)$.

A **divisor** on C is a formal sum of points

$$D = \sum m_P P \tag{6.20}$$

where $m_P = 0$ for all but finitely many points $P \in C$ over $\overline{K}$. The degree of a divisor is $\deg(D) = \sum m_P$, and the support is the set of points $\{P \mid m_P \neq 0\}$. The divisors form a group $\mathrm{Div}(C)$, and the divisors of degree zero form a proper subgroup $\mathrm{Div}^0(C) < \mathrm{Div}(C)$. A **principal** divisor is a divisor of the form $(\gamma) = \sum_{P \in C} \mathrm{ord}_P(\gamma) P$ for some γ in the function field $\overline{\mathbb{F}}_q(C)$. Here, $\mathrm{ord}_P(\gamma)$ is the order of vanishing (the order of the zero, or if negative, the pole) of γ at the point P.

Denote by $\mathcal{P}(C)$ the set of principal divisors of C. It is a fact that principal divisors have degree 0, so $\mathcal{P}(C) < \mathrm{Div}^0(C)$. Then the **divisor class group** or **Jacobian** of C (over $\overline{K}$) is the quotient group

$$\mathrm{Jac}_C \cong \mathrm{Div}^0(C) \Big/ \mathcal{P}(C) \ . \tag{6.21}$$

This is also known as the degree-0 Picard group, denoted by Pic^0. Technically, the Jacobian is an abelian variety, not just a group—but the points of the Jacobian are in one-to-one correspondence with elements of Pic^0, so we treat them as the same. We observe that two divisors $D_1, D_2 \in \mathrm{Div}^0$ are equivalent if $D_1 - D_2 \in \mathcal{P}(C)$.

Computation in the group Jac_C is done with reduced divisors in Mumford representation. It is possible to associate to any element in the group a unique equivalent divisor in $\mathrm{Div}^0(C)$ called a **reduced divisor**. A reduced divisor is one of the form $D = P_1 + \cdots + P_r - r\infty$ (with the P_i finite and not necessarily distinct), with $r \leq g$ and $P_i \neq \tilde{P_j}$ for all $i \neq j$. Reduced divisors have a unique Mumford representation [Mum07] as a pair of polynomials $\langle u(x), v(x) \rangle$, where u is monic, $\deg(v) < \deg(u) \leq g$ (the genus), and $v^2 \equiv f \pmod{u}$ [Can87]. Specifically, the roots of $u(x)$ are the x-coordinates of the points in the support of the divisor.

If we have a curve C of genus g over the finite field of cardinality q, the $\mathbb{F}_q$-rational elements of the Jacobian (that is, the divisor classes where u and v have coefficients in $\mathbb{F}_q$) form a finite group, which we shall denote $\mathrm{Jac}_C(\mathbb{F}_q)$. The Hasse-Weil bound tells us that $\# \mathrm{Jac}_C(\mathbb{F}_q) \sim q^g$, or more precisely,

$$(\sqrt{q} - 1)^{2g} \leq \# \mathrm{Jac}_C(\mathbb{F}_q) \leq (\sqrt{q} + 1)^{2g}. \tag{6.22}$$

An abelian variety—and specifically, in this case, the Jacobian—is **simple** if it does not contain a (non-zero) proper abelian subvariety. For example, if C has a non-trivial map to an elliptic curve E, which induces a map $\mathrm{Jac}_C \to E$, then Jac_C is isogenous to $E \times A$ for some dimension-$(g-1)$ abelian variety A, and $\#A(\mathbb{F}_q)$ divides $\# \mathrm{Jac}_C(\mathbb{F}_q)$. This type of splitting may reduce the difficulty of order computations, an undesirable characteristic for the application we discuss in Chapter 7. Therefore, we want to restrict to simple Jacobians. Fortunately, Jacobians in general are simple with overwhelming probability.

The group law on $\mathrm{Jac}_C(\mathbb{F}_q)$ can be computed using Cantor's algorithm [Can87] (see also the work of Costello and Lauter [CL11]). Efficient explicit formulae exist for $g = 2$ (by Lange [Lan05]) and $g = 3$ (by Fan, Wollinger, and Gong [FWG07]).

On elliptic curves, it is well known that division polynomials $\Psi_n(X)$ exist for all $n \in \mathbb{Z}_+$, where the roots of the n-th division polynomial are the points with order n—that is, the torsion subgroup $E[n]$ minus $\mathcal{O}_E$ (see, for example, [Sil09, Section III] or [Gal12, Section 9.8]). For odd primes ℓ, the degree of Ψ_ℓ is $(\ell - 1)/2$. The hyperelliptic curve analogues of these polynomials are division ideals—homogenous ideals vanishing on the torsion subgroups. In this sense, the genus one division ideals are the principal ideals generated by the n-division polynomials. Comparably, in general genus, there exists a system of equations for each ℓ whose solutions give points or divisors of order ℓ [Can94, CFA$^+$12]. A useful result of Abelard [Abe18] is that in genus three, the degrees of Cantor's ℓ-division polynomials are bounded by $O(\ell^2)$.

Chapter 7

Trustless Groups of Unknown Order

In Chapter 4, we proposed a new zero-knowledge Proof of Knowledge for SIDH secret keys. Continuing now in the theme of zero-knowledge proofs, we turn to a building block relied upon in a number of generic constructions: unknown-order groups.

As the name implies, a group G has **unknown order** (or **hidden order**) if the order of G is infeasible to compute. Of course, to be useful, the group operation on G should be efficiently computable, elements of G should have a compact representation, and it should be possible to efficiently sample random elements of G. Interest in such groups has been fuelled in recent years by a number of interesting applications such as delay functions [BBF18, Wes19], accumulators [BBF19], and zero-knowledge Proofs of Knowledge [BFS20]. Class groups also feature in a number of ECDSA threshold schemes [CCL$^+$19, CCL$^+$21] and multi-signature schemes [CY21].

There are two common settings for these groups, known as **trusted** and **trustless** setup. Trusted setup implies that the order is known to (or computable by) the creator of the group, who has access to extra secret information during the setup of the group. On the other hand, if even the creator(s) of the group G cannot feasibly compute the order of G, the setup is called trustless. We are particularly interested in the trustless setting, which has a number of useful applications but which we believe is under-studied.

Previously, there have been two proposals for concrete unknown-order groups: RSA groups [RSW96], and ideal class groups of imaginary quadratic fields [Lip12, BW88].

RSA groups are groups of the form $(\mathbb{Z}/N\mathbb{Z})^\times$, where $N = pq$ is the product of two primes. Computing the order of $(\mathbb{Z}/N\mathbb{Z})^\times$ is equivalent to factoring N. A trusted party can efficiently generate an RSA modulus that resists all known order-computing attacks (including Sutherland's algorithm, which we shall soon discuss). However, using RSA groups in a trustless setting (where the factorisation of N is not known to anyone) is much more difficult. Sander [San99] gave an algorithm to generate a modulus N in a trustless manner, such that (with very high probability) N has two large factors—he calls this an RSA-UFO (unknown factorisation object). However, to match even the lower security of 1024-bit RSA moduli, RSA-UFOs need "bit length (much) greater than 40,000 bits"; this is far too large to be efficient in most unknown-order group applications.

Class groups, on the other hand, can be generated without a trusted setup, and so have received a lot of recent attention (see, for example, [Lip12, Wes19, BBF19]). Buchmann and Hamdy [BH01] suggested that 1665-bit discriminants ($\approx$ 833-bit orders) provide security equivalent to 3072-bit RSA (i.e., 128-bit security). More recently, Biasse, Jacobson Jr., and Silvester [BJS10] claim that 1827-bit discriminants ($\approx$ 914-bit orders) are required to reach the same security level.

A major result of this chapter is that the usual notions of security level are not appropriate when evaluating class group security for applications such as accumulators, where the group is fixed and used by all participants. In Section 7.2 we recall Sutherland's algorithm for computing the order of a generic group, and reanalyse the security of unknown-order groups with respect to this algorithm. We believe the relevance of this algorithm to the unknown-order group setting has previously been overlooked, and that the recommended discriminant sizes for trustless generation of such groups have been dramatically underestimated.

Brent [Bre00] briefly mentioned using the Jacobian of a hyperelliptic curve as a group of unknown order. This follows work by Koblitz [Kob89] on the use of Jacobians as groups in which the discrete logarithm problem is infeasible. Unlike the use of class groups of imaginary quadratic fields, this Jacobian idea has received very little further attention. The second (and more speculative) contribution of this chapter is to revisit this idea, analysing the use of genus-3 curves as a source of unknown-order groups without trusted setup. This is discussed in Section 7.4, where we propose the idea more concretely.

We find that Jacobians offer a distinct advantage over class groups at the same security level: the element representation size is smaller (2/3 of the size if our new class group compression algorithm from Section 6.3.2 is used, and if not, 1/2 the size), since point compression for curves is optimal. Using Jacobians also allows us to take advantage of the wealth of algorithms for group operations and exponentiation that have been developed and implemented for hyperelliptic discrete-log-based cryptography, which may be more efficient than their class group equivalents (though the lack of recent competitive implementations makes it difficult to compare Jacobians and class groups in terms of real-world speed).

We acknowledge that there are, in theory, polynomial-time algorithms to compute the group order of hyperelliptic Jacobians [Pil90, GH00]. However, there is evidence that these algorithms are already impractical for discrete-log-based cryptographic group orders of around 256-bits, let alone the much larger group orders that we have in mind. While curves of any genus ≥ 2 might be considered, we suggest that genus-3 curves are the best choice: their point-counting algorithms are already very complex, and their DLP is harder relative to higher-genus curves. Naturally, if Schoof-type algorithms for genus three could be made efficient over large prime fields, then these groups would become insecure—but at least we have provided motivation for such work.

Some unknown-order group protocols make stronger assumptions. For example, one may wish to assume that that finding elements of a given order is hard, or that extracting roots of a given element is hard. In Section 7.5 we consider the problem of constructing points of known order in class groups and Jacobians, and explain how we might work with Jacobians when the low-order (LOA) or adaptive root (ARA) assumptions are imposed.

7.1 Motivation: Cryptographic accumulators

To provide some motivation to the idea of unknown-order groups, we shall now briefly outline what a cryptographic **accumulator** is, and why groups of unknown order are useful to create them.

An accumulator is a construction that allows efficient representation of a set of elements in a much smaller (ideally constant) size than the set itself. Accumulators were originally introduced by Benaloh and de Mare [BdM93], and then generalised by Baric and Pfitzmann [BP97]. Given an accumulator value A representing a large set X, it is possible to create a membership witness w for an element $x \in X$ such that anyone with (A, x, w) can verify that x is indeed "in" the accumulator—that is, x is part of the set X which A represents—without needing to provide the entire set X. There are a number of additional properties which accumulators may possess, giving flexibility for use in various situations. For example, a **dynamic** accumulator, introduced by Camenisch and Lysyanskaya [CL02], is one that allows insertion and deletion of accumulated elements—in contrast to the original (**static**) accumulators, which support neither operation (the set X must be fixed at the start). Li, Li, and Xue [LLX07] later proposed **universal** accumulators, which also support proofs of *non*-membership.

Accumulators have attracted attention in recent years due to their potential in blockchain situations, among other uses. Specifically, there have been proposals to represent the current state of the Bitcoin blockchain as a compact dynamic accumulator value, rather than as an unbounded set of unspent transaction outputs (UTXOs) [Dry19, BBF19]. Spending of a UTXO would then require a proof that the UTXO is indeed in the accumulator.

Let us now consider a simple construction. Working in an abelian group G with generator g, suppose we let the "empty" accumulator be $A_0 := g$ (representing an empty set). Given the current state of an accumulator $A_i = g^s$, we can "accumulate" an odd prime p by setting the new accumulator state to $A_{i+1} := A_i^p = g^{s \cdot p}$. The membership of p in A_{i+1} is witnessed by $w_x = A_i$ (the state of the accumulator without x). Membership verification is done by checking that $w_x^p = A_{i+1}$. The requirement that the elements be odd primes is so that, after accumulating two elements p and q, we cannot claim membership of their product pq (or arbitrary combinations of factors from two composite accumulated elements).

To discuss the security of cryptographic accumulators, we require a few important definitions. An accumulator is **complete** if, for any instance of the accumulator A, any element x, and any valid membership (resp., non-membership) proof w_x for $x \in X$ (resp., $x \notin X$), verification of (A, x, w_x) accepts. An accumulator is **sound** if, for any accumulator A representing X, and any element $y \notin X$, it is infeasible for an adversary to generate an accepting proof w_y of membership for y in A. Similarly, for any element $x \in X$, it is infeasible for an adversary to generate an accepting proof of non-membership w_x^* for x in A. Finally, we define undeniability, which is often used as the standard requirement a secure accumulator must satisfy:

Definition 7.1 (Undeniability)**.** An accumulator generated with respect to security parameter κ is undeniable if, for any probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, the probability that $\mathcal{A}$ can produce an accumulator state A, an element x, and both a proof of membership w_x **and** a proof of non-membership w_x^* for x with respect to A is negligible in κ.

Let us consider again the accumulator construction above. An actor who knows the order of g as a group element can efficiently create membership proofs for elements not contained in the accumulator, a violation of the accumulator soundness. This is because computing roots is easy when the group order is known. The original proposals [BdM93, BP97] used the RSA group $(\mathbb{Z}/N\mathbb{Z})^\times$ mentioned above—accumulators constructed in this way are accordingly referred to as RSA accumulators. Clearly, though, anyone who knows the factorisation of the modulus $N = pq$ also knows the order of the group and can forge proofs. Note that a group of unknown order is necessary but not sufficient for security of the RSA accumulator—computing roots may still be easy, and security is usually reduced to the strong RSA assumption in G (which requires that the adversary cannot compute any roots of a given element g). We refer the reader to the work of Boneh, Bünz, and Fisch [BBF19, Theorem 4] for more information.

Lipmaa [Lip12] proposed the static root accumulator as an alternative to the RSA accumulator, without a trusted setup, using groups of unknown order (specifically, the class group of an imaginary quadratic field). The use of unknown-order groups for accumulators was further expanded on by Boneh et al. [BBF19], who propose a dynamic accumulator with efficient membership proof aggregation using groups of unknown order. Trusted setup is an undesirable property in some settings (including the Bitcoin UTXO setting described above)—one reason why unknown-order groups are an interesting avenue of study.

7.2 Sutherland's algorithm: The security of generic groups

Sutherland's **primorial-steps** algorithm [Sut07, Algorithm 4.2] computes the order of an element in a generic group. It can therefore be used to probabilistically determine the exponent of a group. It runs in $O(\sqrt{N/\log\log N}) = o(\sqrt{N})$ time (where N is the group order) in the worst case, but in fact, the expected runtime depends heavily on the multiplicative structure of N. The algorithm runs particularly quickly when N is smooth, which we do not expect (or desire) in unknown-order groups, but it also poses a significant threat to a larger class of groups.

Sutherland's algorithm is based on Shanks' baby-step giant-step (BSGS) algorithm, but one can also use Pollard's rho (explored in Athukorala's thesis [Ath22]). Suppose we wish to compute the order of an element $\alpha \in G$. Instead of computing consecutive powers of α in the baby-steps, we compute a new element $\beta = \alpha^E$ such that the order of β is coprime to all primes $2, 3, \ldots, p_n \leq L$ for a chosen bound L, by taking E to be the product of the p_i, each raised to an appropriate exponent $\lfloor \log_{p_i}(M) \rfloor$ (where M is an upper bound of the group order). The baby-steps are then all powers of β with exponents coprime to P_n, and the giant-step exponents are multiples of P_n, where $P_n = \prod_{i=1}^{n} p_i$. As in BSGS, a collision allows $|\beta|$ to be learned, which then allows $|\alpha|$ to be computed very efficiently.

Sutherland showed that if the order N of α is uniformly distributed over $[1, M]$ (for sufficiently large M) and $L = M^{1/u}$, then this is an $O(M^{1/u})$ time and space algorithm that successfully computes N with probability $P \geq G(1/u, 2/u)$ [Sut07, Proposition 4.7]. Here, $G(r, s)$ is the semi-smooth probability function

$$G(r, s) = \lim_{x \to \infty} \psi(x, x^s, x^r)/x \tag{7.1}$$

for $0 < r < s < 1$, where $\psi(x, y, z)$ is the number of integers up to x that are semi-smooth with respect to y and z—that is, numbers whose prime factors are all less than y, with at most one greater than z.

u	$G(1/u, 2/u)$	u	$G(1/u, 2/u)$	u	$G(1/u, 2/u)$
2.1	0.9488	5.0	0.4473	12.0	4.255e-12
2.9	0.5038	6.0	1.092e-03	16.0	6.534e-19
3.0	0.4473	10.0	5.382e-09	20.0	2.416e-26

Table 7.1: Asymptotic semi-smoothness probabilities from [Sut07] and [BP96].

As mentioned in the introduction of this chapter, we claim that the usual notions of security level are not appropriate when evaluating class group security for applications such as accumulators, where the group is fixed. The computational assumptions underlying security are not defined for a fixed group, and there is no random self-reduction to show that all instances have the same security. We argue that much larger group sizes are needed for secure unknown-order groups in applications where the group is fixed for many users and used for a long period of time.

Precisely, using Sutherland's algorithm as motivation, we propose a new security model for unknown-order groups, depending on two parameters (λ, ρ).

Definition 7.2. Let `Gen` be an algorithm that outputs a group. We say that `Gen` reaches security level (λ, ρ) if, with probability $1 - 1/2^\rho$ over the outputs of `Gen`, any algorithm $\mathcal{A}$ given an output G of `Gen` requires at least 2^λ bit operations to succeed in computing $\#G$ with probability close to 1.

A similar concept of security is implicit in [AMPS18], which considers the security of cryptosystems depending on a prime system-parameter p provided by a possibly malicious party, when in practice the users only verify the primality of p (and thus ensure the security level of the system) up to a certain probability—if at all. One recommendation of [AMPS18] is that users "ensure that composite numbers are wrongly identified as being prime with probability at most 2^{-128}", corresponding to $\rho = 128$.

In our context, the probabilistic nature of security is not due to malicious parties, or unreliable verification, but rather to a fundamental mathematical fact about the distribution of random abelian group orders. The problem is that among class groups with prime discriminants of a given size, there is a set of weak instances *depending on the order*. A randomly-generated group is only vulnerable with small probability, but since the order is unknown, we cannot check for vulnerability without simply attempting to run the algorithm for the given time. In contrast, in the RSA setting with trusted setup, the group order is known to its generator, who can thus easily choose a group that is not vulnerable. Because the runtime of Sutherland's algorithm depends on the (unknown) order itself, rather than the supposed size of the order, we must take into account the non-negligible probability that the order of a generated group can be computed much more efficiently than desired. The relevance of this algorithm to cryptographic unknown-order groups seems to have been overlooked until now.

For each choice of ρ (corresponding to the probability that a weak group is generated), one must determine u such that $1/2^\rho \approx G(1/u, 2/u)$. It then follows that the group size should be at least $u\lambda$ bits so that a $1/u$-th root attack requires at least 2^λ operations. Table 7.1 gives some numerically computed values for $G(1/u, 2/u)$ from [BP96] and [Sut07]. Using the method of Banks and Shparlinski [BS07] to approximate the density of semi-smooth numbers, we calculate that for a success probability of less than 2^{-100}, we should take $u = 22.5$; for 2^{-128}, we should take $u = 26.5$.

Considering all groups generically, we propose new group sizes in response to Sutherland's algorithm, depending on (λ, ρ). A paranoid choice, $(\lambda, \rho) = (128, 128)$, requires group sizes of around 3392 bits, since $u \approx 26.5$ and $26.5 \cdot 128 = 3392$. A more realistic (but still cautious) choice, $(\lambda, \rho) = (128, 55)$, corresponds to $u = 15$ and requires $\approx$ 1920-bit group sizes (and so 3840-bit discriminants; more than double the previous suggestion). Table 7.2 gives sizes of group orders for various combinations of (λ, ρ). An alternative would be to use multiple smaller groups in parallel, however, we believe this approach is less efficient than working in a single larger group. Once again we stress that our setting is different to the usual world of security levels. We are dealing with a fixed class of weak instances of the computational problem.

In the next sections, we will consider the specific cases of class groups and Jacobians, both taking into account Sutherland's algorithm, and the details specific to those constructions.

We remark that Sutherland's algorithm is less of a threat to unknown order groups with trusted setup. For example, if there is an authority that can be trusted to generate an RSA modulus $N = pq$ where p and q are safe primes, then the order of $(\mathbb{Z}/N\mathbb{Z})^\times$ cannot be computed using Sutherland's approach.

		ρ					
		40	55	64	80	100	128
λ	55	660	825	880	1045	1265	1430
	80	960	1200	1280	1520	1840	2080
	100	1200	1500	1600	1900	2300	2600
	128	1536	1920	2048	2432	2944	3392

Table 7.2: Group size (bits) for various attack success probabilities $2^{-\rho}$ and running costs 2^{λ}.

7.3 Ideal class groups as unknown-order groups

In this section, we reconsider the security of imaginary quadratic ideal class groups as a source of trustless unknown-order groups. We assume the background knowledge on ideal class groups that we presented in Section 6.1.

The use of class groups in cryptography was first suggested by Buchmann and Williams [BW88]. Hafner and McCurley [HM89] gave a sub-exponential $L_{|\Delta|}(1/2)$ algorithm for computing the order of $Cl(\Delta)$ in 1989. Buchmann [Buc90] extended this to compute the group structure and discrete logarithms. The important thing to note is that these algorithms all have the same sub-exponential complexity $L_{|\Delta|}(1/2)$, depending essentially on the size of $|\Delta|$.

Thus, the order of a class group $Cl(\Delta)$ of negative prime discriminant $\Delta \equiv 1 \pmod 4$ is believed to be difficult to compute, if Δ is sufficiently large. Lipmaa [Lip12] proposed that $Cl(\Delta)$ can be used as a group of unknown order without trusted setup, simply by selecting a suitably large Δ and choosing an element in $Cl(\Delta)$ to be treated as a generator (it is not possible to know if it generates the whole of $Cl(\Delta)$, or just a subgroup; we discuss this further below). This idea has since been used by Wesolowski [Wes19] and others.

We emphasise that until now, cryptographic class group parameters (that is, the required sizes of the Δ's used) have mainly been proposed with respect to these known sub-exponential algorithms for computing orders of such groups. In this section, we reassess the security of these parameters in light of Sutherland's algorithm and propose new (much larger) parameter sizes targeting the 128-bit security level.

In contrast to the above algorithms, Sutherland's algorithm has exponential worst-case runtime but performs much faster with a non-negligible probability depending on the structure of the class group—a factor that Hafner–McCurley cannot exploit. When computing the order of a random class group, therefore, the small probability that Sutherland's algorithm outperforms Hafner–McCurley must be taken into account.

The cryptographic parameter sizes in [HM00] and [BH01] both suppose that Hafner–McCurley is the best-known algorithm. Concretely, it is suggested that a 1665-bit negative fundamental discriminant, which means an approximately 833-bit group order (by Equation 6.7), should provide 128-bit security. Biasse, Jacobson Jr., and Silvester [BJS10] improve on previous attacks and suggest 1827-bit discriminants (which implies $\sim$ 914-bit orders) are needed to achieve 128-bit security. These estimates have been quoted in more recent works, including by Bünz, Fisch, and Szepieniec [BFS20] who estimate that 1600-bit discriminants provide 120-bit security, and by Boneh, Bünz, and Fisch [BBF19] which proposes a slightly more conservative discriminant size of 2048 bits for 128-bit security.

Suppose we try to compute the order of a random class group with a 1827-bit negative fundamental discriminant using Sutherland's algorithm. Sutherland's algorithm has some important practical speed-ups when specialised from generic groups to class groups—for example, class group element negation is practically free, so time and memory can be reduced by a factor of $\sqrt{2}$ (see [Sut07, Remark 3.1])—but these improvements do not significantly impact security levels. The performance of Sutherland's algorithm on a given quadratic imaginary class group depends entirely on the class number.

Hamdy and Möller [HM00] show that imaginary class numbers are more frequently smooth (although not significantly so) than uniformly random integers of the same size. We may therefore conservatively approximate the smoothness probability of random class group orders as being the same as for random integers. With the results of Section 7.2, the probability that a random class group with 1827-bit fundamental negative discriminant has less than 128 bits of security ($u = 7.1$) is at least $2^{-14.3}$, and the chance it has less than 64-bit security is 2^{-50}. If a system is using a fixed class group as an accumulator, then we need to ask if these probabilities of weakness are acceptable. Such groups do not satisfy Definition 7.2 for $(\lambda, \rho) = (128, 128)$, and so we claim that the security is weaker at these discriminant sizes than was previously thought.

Bach and Peralta [BP96] give $G(1/u, 2/u)$ for $u = 20$ as $2.415504 \times 10^{-26} \approx 2^{-85}$. Thus, even for 85-bit security, we require 3400-bit discriminants. Using Banks and Shparlinski's [BS07] method of approximating $G(1/u, 2/u)$, we estimate that for 100-bit security with respect to Sutherland's algorithm, a discriminant of around 4600 bits would be required. For 128-bit security, $u = 26.5$ gives $G(1/u, 2/u) \approx 2^{-128}$, which implies a group order $N \approx 2^{128\times 26.5} = 2^{3392}$, and hence we estimate that Δ should be approximately 6784 bits. We emphasise that $G(1/u, 2/u)$ is only a lower bound for the success probability of Sutherland's algorithm, but this should still serve at least as a guideline.

7.4 Hyperelliptic Jacobians as unknown-order groups

We now revisit Brent's proposal of using hyperelliptic Jacobians as a concrete source of unknown-order groups, focusing on genus $g = 3$. Hyperelliptic Jacobians can be seen as the ideal class groups of quadratic function fields. We will argue that even despite the existence of theoretical polynomial-time point-counting algorithms, these Jacobians may still present a more efficient alternative to class groups at the same security levels. We assume the background material presented in Section 6.4.

Let C be a hyperelliptic curve of genus g over $\mathbb{F}_q$, where $q = p^n$, and let Jac_C be its Jacobian. Recall that $\# \mathrm{Jac}_C(\mathbb{F}_q) \approx q^3$, from Equation 6.22. For $\mathrm{Jac}_C(\mathbb{F}_q)$ to be useful as an unknown-order group, calculating $\# \mathrm{Jac}_C(\mathbb{F}_q)$ should be infeasible. Besides generic algorithms, two classes of algorithms specific to hyperelliptic Jacobians are relevant here: point-counting and discrete-log algorithms.

As a baseline, C must be chosen such that $\mathrm{Jac}_C(\mathbb{F}_q)$ resists Sutherland's algorithm with acceptable probability, as in Section 7.2. Sutherland's algorithm has some important practical optimisations when specialised to hyperelliptic Jacobians—for example, we can again exploit the fact that negation is effectively free to decrease storage and runtime by a factor of up to $\sqrt{2}$ (see [Sut07, Remark 3.1]), and even if a Jacobian is not directly vulnerable to Sutherland's algorithm, its order may be deduced from that of vulnerable twists, as in [Sut09]. However, these improvements do not significantly impact security levels.

To reach acceptable levels of security against Sutherland's algorithm using genus-3 curves, q must be sub-exponentially large. Looking at Table 7.2, the cautious $(\lambda, \rho) = (128, 55)$ level requires 1920-bit groups, or $q \approx 2^{640}$, while the paranoid $(128, 128)$ level requires 3392-bit groups, or

$q \approx 2^{1131}$. Fields of this size also address the concerns of Lee [Lee20].

7.4.1 Point-counting algorithms

Computing $\#\operatorname{Jac}_C(\mathbb{F}_q)$ is a classic problem (called “point counting”) in algorithmic number theory: the goal is to compute the zeta function of C, from which we immediately get $\#\operatorname{Jac}_C(\mathbb{F}_q)$. The many dedicated point-counting algorithms fall naturally into two broad classes: p-adic algorithms and ℓ-adic “**Schoof-type**” algorithms. The p-adic algorithms (notably Kedlaya’s algorithm [Ked01] and its descendants [Har07]) have complexity polynomial with respect to g and n, but exponential in $\log p$. Taking $q = p$, we can safely ignore these algorithms.

Schoof-type algorithms compute $\#\operatorname{Jac}_C(\mathbb{F}_q)$ in polynomial-time *for fixed g*. Indeed, from a theoretical point of view, the existence of Schoof-type algorithms may make the use of hyperelliptic Jacobians as unknown-order groups seem perverse. But Schoof-type algorithms are totally impractical over large prime fields, even in genus as small as 3. To understand why, we need to look at how they operate.

First, consider the case of elliptic curves ($g = 1$). Schoof’s ground-breaking $\widetilde{O}(\log^5 q)$ algorithm [Sch85], the first polynomial-time point-counting algorithm for elliptic curves, computes the characteristic polynomial of Frobenius for a series of small primes ℓ, using polynomial arithmetic modulo the division polynomials Ψ_ℓ, before combining the results with the Chinese Remainder Theorem (CRT) to compute the trace of Frobenius (and therefore the number of points on the elliptic curve).

Specifically, recall from Equation 1.5 that the (q-power) Frobenius endomorphism $\pi_q : E \to E$ sends points (x, y) to (x^q, y^q). Every endomorphism ϕ satisfies a quadratic equation $\phi^2 - t\phi + d = 0$ in $\operatorname{End}(E)$, where d is the degree of the isogeny and t is called its **trace** [Gal12, Theorem 9.9.3]. This equation is called the characteristic polynomial of the endomorphism. The characteristic polynomial of π_q is

$$\pi_q^2 - [t]\pi_q + [q] = 0, \tag{7.2}$$

where t is the trace of Frobenius we were introduced to in Equation 1.4. Since all points defined over $\mathbb{F}_q$ are fixed by π_q, then $\ker(\pi_q - [1]) = E(\mathbb{F}_q)$. From this it can be derived that $\#E(\mathbb{F}_q) = \deg(\pi_q - [1]) = q + 1 - t$. This gives us the familiar equality $\#E(\mathbb{F}_q) = q + 1 - t$ we have already seen. Thus, by computing t via CRT, we learn the number of points on E.

The successor of Schoof’s algorithm, the Schoof–Elkies–Atkin (SEA) algorithm [Sch95], runs in time $\widetilde{O}(\log^4 q)$, and has made elliptic-curve point counting routine.

Pila generalised Schoof’s algorithm to higher-dimensional abelian varieties [Pil90], including all Jacobians of curves. Pila’s algorithm is polynomial-time in p and n, but badly exponential in g. As far as we know, it has never been implemented. The task gets a little simpler when we specialise from general abelian varieties to hyperelliptic Jacobians. The crucial objects are the analogues of the division polynomials: these are multivariate division ideals vanishing on coordinates of points in torsion subgroups, as we mentioned in Section 6.4. Cantor constructs generators for the ℓ-division ideal in [Can94] (see also the book of Cohen et al. [CFA$^+$12]).

Schoof-type point counting is already challenging in genus two. Several genus two algorithms have been implemented and analysed, beginning with Gaudry and Harley [GH00] and Gaudry and Schost [GS04a]. Pitcher’s PhD thesis [Pit09] gave a genus two algorithm with complexity $O((\log q)^7)$. Gaudry and Schost [GS12] used an improved algorithm, with a mixture of Pitcher’s approach and exponential birthday-paradox algorithms, to find a curve of secure order over the 127-bit Mersenne prime field $\mathbb{F}_p$ with $p = 2^{127} - 1$. In their experiments, they claimed around

1,000 CPU-hours on average to compute the order of a random genus-2 curve over this 127-bit field. Computing ℓ-division ideals and analysing the action of Frobenius on them can become impractical for even moderately small ℓ: the computations mentioned above, with an 8 GB limit on RAM, used primes $\ell \leq 31$ (the earlier [GS04a] used ℓ up to 19). They also used small prime powers $\ell^k = 2^{16}$, 3^6, 5^4, and 7^2. These ℓ and ℓ^k are not sufficient to determine the group order; to finish the order computation, they used one- or two-dimensional random walks (a low-memory algorithm with square root complexity—see [GS04b] for details). The fact that finishing this point-counting computation is a situation where in practice, an exponential algorithm is more practical than a polynomial-time one, underlines the impracticality.

We have found no practical work for general genus-2 curves going beyond $\ell = 31$ in the literature. Abelard's PhD thesis [Abe18] discusses the feasibility of continuing with larger ℓ. With time complexity $\widetilde{O}(\ell^6 \log q)$ and space complexity $\widetilde{O}(\ell^4 \log q)$, running time becomes more of an issue than memory. For 192-bit q, the computation for $\ell = 53$ could take around 1,000 CPU-days, yet still leave a search space of $\sim 2^{95}$ elements in the exponential "collision" step of the algorithm.

This practical work has not been extended to genus three. The main obstruction is the complexity of computing with division ideals. Some theoretical analysis and projected complexities appear in the 2019 work of Abelard, Gaudry, and Spaenlehauer [AGS19b]. First steps were made by Abelard et al. [AGS19a] for the very special class of genus-3 Jacobians with known and efficiently computable real multiplication endomorphisms, following the analogous genus-2 algorithm by Gaudry, Kohel, and Smith [GKS11], but this approach does not apply to general genus-3 hyperelliptic Jacobians.

Concretely, taking $q \sim 2^{100}$ in genus three would appear sufficient to resist point counting on most curves C, and result in $\#\operatorname{Jac}_C(\mathbb{F}_q) \sim 2^{300}$. Therefore, the much larger group and field sizes required to resist Sutherland's algorithm render point counting irrelevant as an attack. While point counting for *fixed* genus $g > 2$ is polynomial-time in theory, it remains impractical—even infeasible—in the real world. This is already true of the relatively small field sizes relevant to discrete-logarithm-based cryptography, and it is even more so for the much larger, sub-exponential-sized fields required to protect against Sutherland's algorithm in the unknown-order setting.

Remark 7.3. Some work has been done on generating genus-2 and genus-3 Jacobians *with a known number of points* using CM theory, for applications in DLP-based cryptography, notably by Weng [Wen01, Wen03] (see also e.g. [GS12] and [HSS01]). Obviously, these curve-generation methods must be avoided for unknown-order applications.

Remark 7.4. One might hope that progress in computing higher-genus modular polynomials might yield a SEA analogue improving substantially on pure Schoof-style point counting. However, any SEA analogue in genus $g > 3$ would actually be *slower* than pure Schoof. Indeed, the number of isogenies splitting $[\ell]$ (and hence the degree of the ideal that a SEA analogue would use at the prime ℓ) is in $O(\ell^{g(g+1)/2})$; this exceeds the degree of the ℓ-division ideal, which is in $O(\ell^{2g})$. Even for $g = 3$, the asymptotic complexity of SEA is no better than that of Schoof.

7.4.2 Discrete logarithm algorithms

If the DLP can be efficiently solved in a subgroup $\langle G \rangle \subset \operatorname{Jac}_C(\mathbb{F}_q)$, the order of $\langle G \rangle$ can also be efficiently computed. Precisely, if $xG = \mathcal{O}$, where $\mathcal{O}$ is the identity element, then x is (a multiple of) the order of $\langle G \rangle$. Suppose, that we want to solve the DLP in $\mathrm{Jac}_C(\mathbb{F}_q)$, where C is a curve of genus g over $\mathbb{F}_q$. Gaudry, Thomé, Thériault, and Diem [GTTD07] and Nagao [Nag04] present algorithms for small g running in time $\widetilde{O}(q^{2-2/g})$, improving on the $O(q^2)$ algorithm of Gaudry [Gau00], and the single-large-prime algorithm of Thériault [Thé03]. This has better performance for genus three than square-root algorithms like Pollard's rho, which has expected runtime in $\widetilde{O}(q^{3/2})$. However, in genus two, Pollard's rho algorithm is more efficient—running in time $\widetilde{O}(q)$. Avanzi, Thériault, and Wang [ATW08] further discuss security in these cases.

Smith [Smi09] gives a method of transferring the DLP from hyperelliptic to non-hyperelliptic genus-3 Jacobians that applies to 18.57% of genus-3 curves. Diem's algorithm [Die06] can then be used to solve the DLP in time $\widetilde{O}(q)$. Laine and Lauter [LL15] examine and improve on Diem's attack (including analysis of the logarithmic factors, which they estimate to be $O(\log^2 q)$), but the memory requirement for their attack is high, at $\widetilde{O}(q^{3/4})$. The practical results from Laine and Lauter's work [LL15] suggest that even for $q \sim 2^{100}$, discrete logarithms require around 2^{113} field multiplications and 1.2×10^{14} TB of memory, assuming the reduction of [Smi09] applies; if not, the algorithm of Gaudry et al. [GTTD07] would require on the order of 2^{133} operations. Genus-3 hyperelliptic curves avoiding isogeny-based attacks are constructed in [Lai15].

As g tends to infinity, there exist sub-exponential attacks on the DLP using index calculus (for example, [Eng02]). However, these have no impact for fixed genus two and three.

7.4.3 Avoiding special curves

Previous work on generating hyperelliptic curves for cryptography focused on generating Jacobians that avoid known DLP attacks. For example, the order should have a large prime factor, to avoid Pohlig–Hellman; the largest prime factor should not divide $q^k - 1$ for small k, to avoid MOV-type attacks [FR94]; and the group order should be prime to $p = \mathrm{char}(\mathbb{F}_q)$ to avoid "anomalous curve" attacks [Rüc99]. In the context of unknown-order groups, it is (by definition) not possible to know whether the Jacobian meets these conditions or not. Fortunately, the vulnerable group orders are extremely rare: a randomly generated hyperelliptic Jacobian will have a large prime dividing its order with very high probability. The security of random ideal class groups as groups of unknown order depends on similar assumptions and heuristics [CL84].

To ensure that we do not reduce the difficulty of point counting using maps to subvarieties (as mentioned in Section 6.4), the curve should have a simple Jacobian—that is, there should not be any non-trivial abelian subvarieties in the Jacobian. A randomly chosen C will ensure this with overwhelming probability (generic Jacobians are absolutely simple), but we should still be careful to ensure that there is no morphism $C \to D$ with D not isomorphic to C or the projective line (for example, D an elliptic curve), since then Jac_D would be a non-trivial abelian subvariety of Jac_C.

We must also exclude curves whose Jacobians have special endomorphisms, such as the efficiently-computable real multiplication exploited by Abelard et al. [AGS19a]. Again, a randomly chosen C will avoid these special classes of curves with overwhelming probability, since they form positive-codimensional subspaces of the moduli space. Recent work of Thakur [Tha20] further discusses classes of curves to avoid. The thesis of Alexandrovich [Ale22] also suggests classes of curves to avoid and potential mechanisms for generating safe curves. We shall not delve into such details here, but instead refer the interested reader to the works mentioned.

7.4.4 Generating hyperelliptic Jacobians of unknown order

We now concretely propose a simple method for generating Jacobians of genus-3 hyperelliptic curves, for use as groups of unknown order. Algorithm 7.1 (`Gen`) takes security parameters (λ, ρ) (as in Definition 7.2), and outputs a generator P for a group G such that Sutherland's algorithm, running in time 2^{λ}, succeeds in computing $\#G$ with probability less than $1/2^{\rho}$. The group G is realised as (a subgroup of) a genus-3 hyperelliptic Jacobian. Having chosen a suitable prime p as a function of (λ, ρ), the algorithm samples a uniformly random monic irreducible degree-7 polynomial $f(x)$ in $\mathbb{F}_p[x]$ and polynomials u and v such that $\langle u, v\rangle$ is the Mumford representation of a divisor class P in $\mathrm{Jac}_C(\mathbb{F}_p)$, where C is the curve defined by $y^2 = f(x)$. Being random, P generates a large-order subgroup of $\mathrm{Jac}_C(\mathbb{F}_p)$ with high probability.

Taking f to be random makes the probability that C is a "weak" curve overwhelmingly small, as mentioned above. Furthermore, taking f irreducible over $\mathbb{F}_p$ ensures that $\mathrm{Jac}_C(\mathbb{F}_p)$ has no points of order 2. As we will see in Section 7.5, it may be possible to construct points of small odd order. We could try this for a few small primes ℓ to eliminate C with small factors in $\#\,\mathrm{Jac}_C(\mathbb{F}_p)$, but this makes no significant difference to the probability of semi-smoothness of $\#\,\mathrm{Jac}_C(\mathbb{F}_p)$, and thus to the effectiveness of Sutherland's algorithm. Our simulations showed that rejecting random group orders divisible by the first few primes decreased the semi-smoothness probability by less than a factor of two.

Algorithm 7.1 `Gen`. Constructs a random unknown-order (subgroup of a) genus-3 hyperelliptic Jacobian.

```
Input: $(\lambda, \rho)$
Output: A prime $p$, a hyperelliptic genus-3 curve $C/\mathbb{F}_p$, and $P \in \mathrm{Jac}_C(\mathbb{F}_p)$ such that $\langle P\rangle$ has unknown order
Determine $n$ such that a random genus-3 curve over an $n$-bit prime field has $\lambda$-bits of security
    with probability $1 - 1/2^{\rho}$
$p \leftarrow$ a random $n$-bit prime
Sample random $u(x) = x^3 + u_2x^2 + u_1x + u_0$ in $\mathbb{F}_p[x]$
Sample random $v(x) = v_2x^2 + v_1x + v_0$ in $\mathbb{F}_p[x]$
repeat
    Sample random $w(x) = x^4 + w_3x^3 + w_2x^2 + w_1x + w_0$ in $\mathbb{F}_p[x]$
    $f(x) := v(x)^2 + u(x)w(x)$
until $\gcd(f(x), f'(x)) = 1$ and $f$ is irreducible
$P \leftarrow \langle u, v\rangle$
return $(p, C, P)$ where $C$ is the hyperelliptic curve $y^2 = f(x)$ over $\mathbb{F}_p$
```

To ensure that not even the constructor of C knows $\#\,\mathrm{Jac}_C(\mathbb{F}_p)$, and that C and $P = \langle u, v\rangle$ were indeed generated randomly, we suggest that u, v, and w be chosen by a deterministic "nothing up my sleeve"-type process. For example, the coefficients of the polynomials might be taken from the hash of a certain string. Suppose this process was manipulated by taking multiple "seeds", and testing each resulting curve for weakness. If the probability of encountering a weak curve among random curves is δ, and testing for weakness costs 2^n operations, then a malicious actor requires around $2^n/\delta$ operations to generate a weak C. A sceptical verifier, on the other hand, must only test the proposed C just once to detect cheating, at a cost of just 2^n operations. This imbalance between the cost of cheating versus verifying is a deterrent for attackers, regardless of the weakness in question.

Now, the order of the Jacobian $\mathrm{Jac}_C(\mathbb{F}_p)$ (and the subgroup generated by P) cannot feasibly be computed, not even by the party who constructed the curve. We have therefore achieved a trustless setup, as desired. This group can then be used in cryptographic constructions including accumulators and verifiable delay functions (VDFs). Overall, the generation of a new hyperelliptic curve is relatively cheap. Therefore, just as in the case of class groups, it should be feasible to generate a new group of unknown order for each new instance of an accumulator or VDF if desired.

Elements of $\mathrm{Jac}_C(\mathbb{F}_q)$ are represented as pairs of polynomials $\langle u, v\rangle$ with $\deg(v) < \deg(u) \leq g$, so elements can be stored concretely with six elements of $\mathbb{F}_q$, and further compressed to just three $\mathbb{F}_q$-elements and three extra bits using the method of Hess, Seroussi, and Smart [HSS01]. For $(\lambda, \rho) = (128, 55)$, with $\sim$ 640-bit fields, this means that group elements can be stored in approximately 1920 bits. On the other hand, elements of a class group of equivalent security require $\sim$ 2880 bits with the compression of Section 6.3, or around 3840 bits without it. Moving to the more paranoid security level of $(\lambda, \rho) = (128, 128)$, hyperelliptic Jacobian elements require $\sim$ 3396 bits while class group elements require $\sim$ 5090 bits (or $\sim$ 6784 bits without the compression of Section 6.3). We therefore claim that genus-3 Jacobians offer more compact elements than class groups at the same security level.

Given that hyperelliptic Jacobians are a function-field analogue of ideal class groups of quadratic fields, it is natural to ask why the almost-ideal hyperelliptic Jacobian element compression algorithm of Hess et al. [HSS01] does not have an efficient class group analogue—why is the compression algorithm given in Section 6.3 not as efficient as this? To compress a Jacobian element $\langle u, v\rangle$, the algorithm of Hess et al. [HSS01] begins by factoring u, a polynomial over a finite field. This can be done efficiently. However, a class group analogue compressing (a, b) would need to factor the integer a, which is a much harder problem, as we know. The algorithm we presented in Section 6.3 avoids this issue and is efficient to use.

We conclude this section with Tables 7.3 and 7.4, comparing the relative sizes of parameters and element representations of ideal class groups and Jacobians, for different security parameters (λ, ρ).

		ρ					
		40	55	64	80	100	128
λ	55	$q = 220$ $1320, 663$	$q = 275$ $1650, 828$	$q = 293$ $1758, 882$	$q = 348$ $2088, 1047$	$q = 422$ $2532, 1269$	$q = 477$ $2862, 1434$
	80	$q = 320$ $1920, 963$	$q = 400$ $2400, 1203$	$q = 427$ $2562, 1284$	$q = 507$ $3042, 1524$	$q = 613$ $3678, 1842$	$q = 693$ $4158, 2082$
	100	$q = 400$ $2400, 1203$	$q = 500$ $3000, 1503$	$q = 533$ $3198, 1602$	$q = 633$ $3798, 1902$	$q = 767$ $4602, 2304$	$q = 867$ $5202, 2604$
	128	$q = 512$ $3072, 1539$	$q = 640$ $3840, 1923$	$q = 683$ $4098, 2052$	$q = 781$ $4686, 2346$	$q = 981$ $5886, 2946$	$q = 1131$ $6786, 3396$

Table 7.3: Parameter q, uncompressed element size, and compressed element size (respectively) using Jacobians of genus-3 curves at various security levels.

7.5 Elements of known order

We now briefly consider the problem of constructing points of known order in groups of unknown order. As mentioned at the start of this chapter, a number of settings require it to be infeasible to compute elements of low or known order, or to find roots of elements in the group. A better

		ρ					
		40	55	64	80	100	128
λ	55	$\Delta = 1320$ $1320, 990$	$\Delta = 1650$ $1650, 1238$	$\Delta = 1758$ $1758, 1319$	$\Delta = 2088$ $2088, 1566$	$\Delta = 2532$ $2532, 1899$	$\Delta = 2862$ $2862, 2147$
	80	$\Delta = 1920$ $1920, 1440$	$\Delta = 2400$ $2400, 1800$	$\Delta = 2562$ $2562, 1922$	$\Delta = 3042$ $3042, 2282$	$\Delta = 3678$ $3678, 2759$	$\Delta = 4158$ $4158, 3119$
	100	$\Delta = 2400$ $2400, 1800$	$\Delta = 3000$ $3000, 2250$	$\Delta = 3198$ $3198, 2399$	$\Delta = 3798$ $3798, 2849$	$\Delta = 4602$ $4602, 3452$	$\Delta = 5202$ $5202, 3902$
	128	$\Delta = 3072$ $3072, 2304$	$\Delta = 3840$ $3840, 2880$	$\Delta = 4098$ $4098, 3074$	$\Delta = 4686$ $4686, 3515$	$\Delta = 5886$ $5886, 4415$	$\Delta = 6784$ $6784, 5090$

Table 7.4: Parameter Δ, uncompressed element size, and compressed element size (respectively) using ideal class groups at various security levels.

understanding of points of known order will give us an idea of how to work with Jacobians when these low-order or adaptive root assumptions are imposed.

7.5.1 Low-order assumptions and cofactors

We now define some of the common additional requirements on unknown-order groups more concretely:

- The **low-order assumption** states that finding an element P and its order $1 < s < 2^{\kappa}$ in G (for security parameter κ) should be computationally infeasible (see [BBF18, Definition 1]).
- The **adaptive root assumption** states that computing random (prime) roots of any non-trivial element in the group should be computationally infeasible. That is, it is infeasible for any efficient adversary to output a non-trivial element Q such that, given a random prime ℓ, the adversary also outputs P such that $Q = [\ell]P$ in G (see [BBF18, Definition 2] and [Wes19]).

We emphasise that these assumptions only make sense if the adversary must solve arbitrary instances in a randomly-sampled G. It is not possible to define security for a fixed G.

To motivate these assumptions, we now outline Wesolowski's Proof of Exponentiation (PoE), a protocol for proving that a value w is the correctly-computed x-th power of another element u, without requiring the verifier to repeat the full exponentiation computation. We present this protocol in Example 7.5.

Example 7.5 (Proof of Exponentiation (PoE))**.** Let G be a group, chosen according to security parameter κ. The Proof of Exponentiation takes as input u and w in G and $x \in \mathbb{Z}$, and aims to prove that $u^x = w$ in significantly less time than it takes to compute u^x. It proceeds interactively as follows:

1. Verifier sends a random prime $\ell \leftarrow \mathsf{Primes}(\kappa)$ to the prover.
2. Prover computes $q = \lfloor x/\ell \rfloor$ and $Q = u^q$, and sends Q to the verifier.
3. Verifier computes $r = x \bmod \ell$, and accepts if $Q^{\ell} u^r = w$.

Here, $\mathsf{Primes}(\kappa)$ is a set containing the first 2^{κ} prime numbers. This protocol can be made non-interactive with the Fiat–Shamir transformation, by hashing u, w into $\mathsf{Primes}(\kappa)$ (see [Wes19]).

To see why the security of this protocol requires the low-order assumption, suppose we know an element ϵ of order 2 in G (for example, if G is an RSA group, then we can take $\epsilon = -1$). Then for any valid proof that $u^x = w$, we can easily generate a false proof of the contradictory statement $u^x = \epsilon w$, by replacing Q with $Q' = \epsilon Q$ in the proof. Since ℓ is odd, $(Q')^{\ell} u^r = \epsilon Q^{\ell} u^r = \epsilon w$ holds despite the fact that $u^x \neq \epsilon w$. This is why, when using RSA groups, it is important to use the quotient $(\mathbb{Z}/N\mathbb{Z})^{\times}/\langle \pm 1 \rangle$ to eliminate this element.

Furthermore, if the prover is able to compute arbitrary roots of elements in G, then in Step 2 of the PoE protocol they can compute an ℓ-th root of $\tilde{w}/u^r$ for any value $\tilde{w}$ of their choice, and wrongfully convince the verifier that $u^x = \tilde{w}$. Hence, the adaptive root assumption in G is also required for this protocol to be sound.

Suppose we are given an algorithm `Gen` constructing unknown-order groups reaching the (λ, ρ) security level. Suppose too that $\mathcal{S}$ is a set containing all the integers s such that we can construct elements of order s or extract s-th roots in groups G output by `Gen` in $< 2^{\lambda}$ operations with probability $> 2^{-\rho}$. If we can specify such a set $\mathcal{S}$, then the low-order and adaptive root assumptions hold in the subgroup

$$[S]G = \{[S]P \mid P \in G\} \qquad \text{where} \qquad S := \operatorname{lcm}(\mathcal{S}) \,. \tag{7.3}$$

This cofactor S "kills off" all elements of small order. By definition, it also raises all elements by a multiple of every integer s, making the ability to find s-th roots in G useless. We will propose conservative choices for $\mathcal{S}$ for concrete groups below. In the meantime, to give some concrete intuition, if we take $\mathcal{S} = \{1, \ldots, 60\}$ then S is an 84-bit integer, so multiplication by S is efficient.

The operation of protocols such as accumulators in $[S]G$ is standard, but some protocols may need modification. For example, proofs may require an extra check that an element is indeed in the group $[S]G$. The issue here is that, given a point Q in G, testing subgroup membership $Q \in [S]G$ is not easy. However, the original point Q is effectively a proof that $[S]Q$ is in $[S]G$, because this can be easily verified. Consequently, Q should be sent instead of $[S]Q$ in cryptographic protocols, and the verifier can perform the multiplication by S themselves.

Using $[S]G$ in place of G has an impact on efficiency, due to the extra scalar multiplications required. This impact is highly protocol-dependent, but in most cases only a few extra multiplications should be needed. To give a specific example, we revisit the PoE from Example 7.5 in Example 7.6 (note that we use additive rather than multiplicative notation now for clarity). The verifier only needs to perform one extra multiplication-by-S during verification when working in $[S]G$ instead of G. We suggest that this is efficient enough for practical use, and that other protocols using the adaptive root assumption can be modified in a similar way.

Example 7.6 (PoE in $[S]G$)**.** We begin the PoE protocol with input $U \in G$, $W \in [S]G$, and $x \in \mathbb{Z}$. The claim to be proven is that $[x][S]U = W$ in $[S]G$. The protocol proceeds as follows:

1. Verifier selects a random ℓ from $\text{Primes}(\lambda) \setminus \mathcal{S}$ and sends ℓ to the prover.
2. Prover computes $q = \lfloor x/\ell \rfloor$, computes $Q = [q]U$ in G and sends Q to the verifier.
3. Verifier computes $r = x \bmod \ell$, and accepts if Q is in G and $[S]([\ell]Q + [r]U) = W$.

The security of this protocol depends on the choice of $\mathcal{S}$. Given a valid proof of $[x][S]U = W$, in order to falsely prove $[x][S]U = W + P$, the prover must compute $[1/\ell]P$ for the ℓ chosen by the verifier. This may be possible if the order of P is known, but this is supposed to be infeasible because ℓ is not in $\mathcal{S}$.

Remark 7.7. The impact of finding small-order elements is highly domain-specific. For example, in the VDF of [Wes19, BBF18], even if points of known order can be found, forging a false PoE still requires knowing the true result of the exponentiation—and hence still requires computing the output of the VDF. Relying on a PoE would thus break the requirement that the VDF output is unique, but it would still provide assurance of the delay. For accumulators, we need an analogue of the strong RSA assumption rather than the adaptive root assumption: it should be hard to find *chosen* prime roots of an element (recall that the membership witness of ℓ in A is the ℓ-th root of A). This case can be addressed differently, by simply disallowing the accumulation of small primes ℓ dividing elements of $\mathcal{S}$. Finding ℓ-th roots with ℓ not in $\mathcal{S}$ is supposed to be infeasible, so here we do not need to use $[S]G$.

7.5.2 Elements of known order in class groups and Jacobians

For class groups, it is well-known that the factorisation of Δ reveals the 2-torsion structure of $Cl(\Delta)$, and even allows the explicit construction of elements of order 2. Similarly, for Jacobians, if $C/\mathbb{F}_q$ is defined by $y^2 = f(x)$, then the factorisation of $f(x)$ reveals the 2-torsion structure of $\mathrm{Jac}_C(\mathbb{F}_q)$, and lets us construct explicit points of order 2. This motivates the restriction to negative prime Δ when using $Cl(\Delta)$ as an unknown-order group, and our restriction to irreducible f in Algorithm 7.1.

Belabas, Kleinjung, Sanso, and Wesolowski [BKSW20] give several constructions of special discriminants Δ together with a known ideal of small odd order in $Cl(\Delta)$. Similarly, we can construct hyperelliptic Jacobians equipped with a point of small order, as shown by Dobson, Galbraith, and Smith [DGS21]. These discriminants and curves generally do not occur when Δ or f is chosen in a "nothing up my sleeve" way. In any case, the risk of choosing groups with constructible small-order elements can be eliminated by using a smooth cofactor S.

There are three curve-specific methods for constructing elements of known small order, or deducing information about small divisors of the order of a given element, which do not apply to class groups. The first is to use the division ideals. This is practical for small primes like 2, 3, and 5 (reinforcing the need for the cofactor S above). However, if we assume that there exists no feasible Schoof-type algorithm for counting points on genus-3 curves, then we implicitly assume that it is infeasible to construct ℓ-division ideals for ℓ larger than some bound that is polynomial with respect to the security parameters.

The second method is to use repeated divisions by 2 in $\mathrm{Jac}_C(\mathbb{F}_q)$ to construct points of order 2^k for arbitrarily large k. Since 2^k is coprime to all odd primes ℓ, this allows a malicious prover to easily find ℓ-th roots for these points (that is, given a point Q, find P such that $[\ell]P = Q$). But repeated division by 2 in $\mathrm{Jac}_C(\mathbb{F}_q)$ requires the repeated extraction of square roots in $\mathbb{F}_p$, which quickly requires repeatedly taking quadratic field extensions, and the field computations blow up exponentially. Using hyperelliptic curves in the form $y^2 = f(x)$ with $f(x)$ irreducible ensures that the required square roots do not exist in $\mathbb{F}_p$.

Similarly, we might calculate repeated divisions by very small odd primes. Using the group $[S]\,\mathrm{Jac}_C(\mathbb{F}_q)$ will kill off powers of these small primes dividing the group order. It could also be possible to simply test for these repeated divisions during the curve generation procedure, allowing parties to check for small factors of the group order—and then kill these off with a tailored choice of S. It is an interesting open problem to generate an easily verifiable proof that a Jacobian does not have any points of low order.

The third method involves the Tate pairing, another pairing similar to the Weil pairing seen earlier in this thesis. Let C be a hyperelliptic curve over $\mathbb{F}_q$, let ℓ be a prime (coprime to q), and let k be the smallest positive integer such that $\ell \mid q^k - 1$. The reduced ℓ-Tate pairing is a bilinear mapping

$$t_\ell : \mathrm{Jac}_C[\ell] \times \mathrm{Jac}_C(\mathbb{F}_{q^k}) \Big/ \ell \,\mathrm{Jac}_C(\mathbb{F}_{q^k}) \longrightarrow \boldsymbol{\mu}_\ell \subset \mathbb{F}_{q^k}^\times$$

where $\boldsymbol{\mu}_\ell$ is the group of ℓ-th roots of unity (see [GHV07] for background on pairings on hyperelliptic curves). If we can find points of known order ℓ, then the ℓ-Tate pairing can give information about the ℓ-divisibility of other points.

Suppose we can find a point Q in $\mathrm{Jac}_C(\mathbb{F}_{q^k})$ of known small-prime order ℓ. Then, for any point P in $\mathrm{Jac}_C(\mathbb{F}_q)$, we can efficiently compute $t_\ell(Q, P)$ in $\boldsymbol{\mu}_\ell$ (assuming k is only polynomially large in $\log q$). Now, if $\ell \nmid |P|$ (the order of P), then $P = \ell P'$ for some P', so $t_\ell(Q, P) = 1$. By the contrapositive, if $t_\ell(Q, P) \neq 1$, then ℓ divides the order of P.

Unfortunately, the converse is not so simple: $t_\ell(Q, P) = 1$ for a single point Q of order ℓ does not imply $\ell \nmid |P|$. Instead, it must be shown that $t_\ell(Q, P) = 1$ for *all* Q in $\mathrm{Jac}_C[\ell]$. Thus, we require a basis $\{Q_1, \ldots, Q_{2g}\}$ of $\mathrm{Jac}_C[\ell]$ which we can test: if $t_\ell(Q_i, P) = 1$ for $1 \leq i \leq 2g$, then the bilinearity of the Tate pairing implies $t_\ell(Q, P)$ for all Q in $\mathrm{Jac}_C[\ell]$, and hence that $\gcd(|P|, \ell) = 1$.

The utility of this approach is limited by the difficulty of constructing points of order ℓ, but also by the field extension degree k (since the coordinates of Q and the value of $t_\ell(Q, P)$ are in $\mathbb{F}_{q^k}$). The embedding degree k, being the order of q modulo ℓ, tends to blow up with ℓ. If q is well-chosen, then in practice we can learn very little information about the orders of random points in $\mathrm{Jac}_C(\mathbb{F}_q)$, or any information at all for points in $[S]\,\mathrm{Jac}_C(\mathbb{F}_q)$ for a suitable S.

For the Jacobian case, we conjecture that $\mathcal{S} = \{1, \ldots, 60\}$ is sufficient for a $(128, 128)$ security level, based on the discussion in Section 7.4.1. For class groups, $\mathcal{S}$ can either be empty in the case of a prime discriminant, or $\mathcal{S} = \{2\}$ if a non-prime discriminant is used.

Conclusions and Future Work

We thank the reader for the time they have taken in order to make it to the end of this thesis. We hope the experience was worthwhile.

This thesis has covered a varied range of topics, from post-quantum key exchange protocols, to classical groups of unknown order; analyses of hardness assumptions, to Proofs of Knowledge. Through it all, we hope that the usefulness and beauty of the mathematics involved—elliptic curves, isogenies, hyperelliptic curves, pairings, and more—has been visible and fascinating. It is our goal that those familiar with these topics have found something new, and those less familiar have been introduced and motivated.

We also hope that the work in this thesis will be built and improved upon, to develop even better schemes and protocols. Progress is of benefit to everyone. We therefore finish this thesis with a brief summary of some ideas for future pursuits, with the goal of encouraging new ideas and developments in these areas and beyond.

The landscape of SIDH-based hardness assumptions we touched on in Chapter 2 is complex and diverse. It would be valuable work to more rigorously analyse these assumptions (especially the new assumptions we introduce). A more standard, formal documentation of these assumptions, which come in different forms and under different names across the literature, would also be useful. SIDH-based assumptions seem to resist easy reductions to other, existing problems, even when they appear strongly related. Thus, any progress toward proving new reductions between these problems would be a relief.

As mentioned in Chapter 4, we expect that further improvements to the efficiency and size of the SIDH zero-knowledge proof and identification schemes presented in that chapter are possible with more analysis. At the end of that chapter (in Section 4.4.1), we outlined some ideas for more efficient protocols, which certainly need much more thought and study. This could be an interesting and worthwhile direction of research, as better efficiency of the SIDH PoK would be an important contribution for practical applications.

In Chapter 5, we proposed a post-quantum replacement for a classical, real-world protocol—Signal X3DH key agreement. While the theoretical development of new schemes and primitives is vital in cryptography, applying these primitives in concrete protocols is also important and necessary. There are a large number of protocols being used on the internet today with only classical security, so there are plenty of inviting problems to work on in developing efficient post-quantum alternatives for these systems—including further improvements to the Signal protocol beyond what we have proposed in this thesis.

Regarding groups of unknown order, we believe our proposal of generating these groups from Jacobians of hyperelliptic curves (in Chapter 7) is worthy of more consideration and study. It would be an especially interesting direction to work further on efficient point-counting algorithms for genus-3 curves. In general, we believe that unknown-order groups are under-studied, and more analysis of both Jacobians and ideal class groups in the light of assumptions such as the low-order and adaptive root assumptions would be worthwhile—as would be examination of other ways in which the trustless construction process for these groups could be manipulated (or proofs that it has not been).

Finally, the required sizes we propose in Chapter 7 for the orders of secure unknown-order groups are much larger than previously suggested parameters. Thus, further work on improving the efficiency of operating in large class groups and Jacobians would also be valuable.

Bibliography

[AAA$^+$20] Gorjan Alagic, Jacob Alperin-Sheriff, Daniel Apon, David Cooper, Quynh Dang, John Kelsey, Yi-Kai Liu, Carl Miller, Dustin Moody, Rene Peralta, et al., *Status report on the second round of the NIST post-quantum cryptography standardization process*, US Department of Commerce, NIST (2020).

[Abe18] Simon Abelard, *Counting points on hyperelliptic curves in large characteristic: Algorithms and complexity*, Ph.D. thesis, Université de Lorraine, 2018.

[ACC$^+$17] Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, David Jao, Brian Koziel, Brian A. LaMacchia, Patrick Longa, et al., *Supersingular isogeny key encapsulation*, Submission to the NIST Post-Quantum Standardization project (2017).

[ACD19] Joël Alwen, Sandro Coretti, and Yevgeniy Dodis, *The double ratchet: Security notions, proofs, and modularization for the Signal protocol*, Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I, Lecture Notes in Computer Science, vol. 11476, Springer, 2019, pp. 129–158.

[AGS19a] Simon Abelard, Pierrick Gaudry, and Pierre-Jean Spaenlehauer, *Counting points on genus-3 hyperelliptic curves with explicit real multiplication*, Proceedings of the Thirteenth Algorithmic Number Theory Symposium, Open Book Ser., vol. 2, Math. Sci. Publ., Berkeley, CA, 2019, pp. 1–19.

[AGS19b] ______, *Improved complexity bounds for counting points on hyperelliptic curves*, Found. Comput. Math. **19** (2019), no. 3, 591–621.

[AJL17] Reza Azarderakhsh, David Jao, and Christopher Leonardi, *Post-quantum static–static key agreement using multiple protocol instances*, Selected Areas in Cryptography - SAC 2017 - 24th International Conference, Ottawa, ON, Canada, August 16-18, 2017, Revised Selected Papers, 2017, pp. 45–63.

[Ajt96] Miklós Ajtai, *Generating hard instances of lattice problems (extended abstract)*, Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996, ACM, 1996, pp. 99–108.

[Ale22] Novoselov Semen Alexandrovich, *Подсчёт числа точек на гиперэллиптических кривых с геометрически разложимым якобианом [Counting the number of points on hyperelliptic curves with a geometrically decomposable Jacobian]*, Thesis, Immanuel Kant Baltic Federal University, 2022.

[AMPS18] Martin R. Albrecht, Jake Massimo, Kenneth G. Paterson, and Juraj Somorovsky, *Prime and prejudice: Primality testing under adversarial conditions*, Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018, ACM, 2018, pp. 281–298.

[Ath22] Pabasara Athukorala, *Point counting on groups of unknown order*, Master's thesis, The University of Auckland, 2022, unpublished.

[ATW08] Roberto Avanzi, Nicolas Thériault, and Zheng Wang, *Rethinking low genus hyperelliptic Jacobian arithmetic over binary fields: Interplay of field arithmetic and explicit formulæ*, J. Math. Cryptol. **2** (2008), no. 3, 227–255.

[BBC$^+$21] Gustavo Banegas, Daniel J. Bernstein, Fabio Campos, Tung Chou, Tanja Lange, Michael Meyer, Benjamin Smith, and Jana Sotáková, *CTIDH: Faster constant-time CSIDH*, IACR Trans. Cryptogr. Hardw. Embed. Syst. **2021** (2021), no. 4, 351–387.

[BBF18] Dan Boneh, Benedikt Bünz, and Ben Fisch, *A survey of two verifiable delay functions*, Cryptology ePrint Archive, Report 2018/712, 2018, `https://ia.cr/2018/712`.

[BBF19] ______, *Batching techniques for accumulators with applications to IOPs and stateless blockchains*, Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part I, Lecture Notes in Computer Science, vol. 11692, Springer, 2019, pp. 561–586.

[BBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev, *Scalable, transparent, and post-quantum secure computational integrity*, Cryptology ePrint Archive, Report 2018/046, 2018, `https://ia.cr/2018/046`.

[BdM93] Josh Cohen Benaloh and Michael de Mare, *One-way accumulators: A decentralized alternative to digital signatures (extended abstract)*, Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings, Lecture Notes in Computer Science, vol. 765, Springer, 1993, pp. 274–285.

[Beu22] Ward Beullens, *Breaking Rainbow takes a weekend on a laptop*, Cryptology ePrint Archive, Report 2022/214, 2022, `https://ia.cr/2022/214`.

[BFG$^+$20] Jacqueline Brendel, Marc Fischlin, Felix Günther, Christian Janson, and Douglas Stebila, *Towards post-quantum security for Signal's X3DH handshake*, Selected Areas in Cryptography - SAC 2020 - 27th International Conference, Halifax, NS, Canada (Virtual Event), October 21-23, 2020, Revised Selected Papers, Lecture Notes in Computer Science, vol. 12804, Springer, 2020, pp. 404–430.

[BFG$^+$22] Jacqueline Brendel, Rune Fiedler, Felix Günther, Christian Janson, and Douglas Stebila, *Post-quantum asynchronous deniable key exchange and the Signal handshake*, Public-Key Cryptography - PKC 2022 - 25th IACR International Conference on Practice and Theory of Public Key Cryptography, Virtual Event, March 8-11, 2022, Proceedings, Part II, Lecture Notes in Computer Science, vol. 13178, Springer, 2022, pp. 3–34.

[BFS20] Benedikt Bünz, Ben Fisch, and Alan Szepieniec, *Transparent SNARKs from DARK compilers*, Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part I, Lecture Notes in Computer Science, vol. 12105, Springer, 2020, pp. 677–706.

[BH01] Johannes Buchmann and Safuat Hamdy, *A survey on IQ cryptography*, Public Key Cryptography and Computational Number Theory, De Gruyter Proceedings in Mathematics, De Gruyter, 2001, pp. 1–15.

[BJM97] Simon Blake-Wilson, Don Johnson, and Alfred Menezes, *Key agreement protocols and their security analysis*, Cryptography and Coding, 6th IMA International Conference, Cirencester, UK, December 17-19, 1997, Proceedings, Lecture Notes in Computer Science, vol. 1355, Springer, 1997, pp. 30–45.

[BJS10] Jean-François Biasse, Michael J. Jacobson Jr., and Alan K. Silvester, *Security estimates for quadratic field based cryptosystems*, Information Security and Privacy - 15th Australasian Conference, ACISP 2010, Sydney, Australia, July 5-7, 2010. Proceedings, Lecture Notes in Computer Science, vol. 6168, Springer, 2010, pp. 233–247.

[BKM$^+$20] Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Charlotte Weitkämper, *On adaptive attacks against Jao–Urbanik's isogeny-based protocol*, Progress in Cryptology - AFRICACRYPT 2020 - 12th International Conference on Cryptology in Africa, Cairo, Egypt, July 20-22, 2020, Proceedings, Lecture Notes in Computer Science, vol. 12174, Springer, 2020, pp. 195–213.

[BKP20] Ward Beullens, Shuichi Katsumata, and Federico Pintore, *Calamari and Falafl: Logarithmic (linkable) ring signatures from isogenies and lattices*, Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part II, Lecture Notes in Computer Science, vol. 12492, Springer, 2020, pp. 464–492.

[BKSW20] Karim Belabas, Thorsten Kleinjung, Antonio Sanso, and Benjamin Wesolowski, *A note on the low order assumption in class group of an imaginary quadratic number fields*, Cryptology ePrint Archive, Report 2020/1310, 2020, `https://ia.cr/2020/1310`.

[BKV19] Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren, *CSI-FiSh: Efficient isogeny based signatures through class group computations*, Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part I, Lecture Notes in Computer Science, vol. 11921, Springer, 2019, pp. 227–247.

[Ble04] Daniel Bleichenbacher, *Compressing Rabin signatures*, Topics in Cryptology - CT-RSA 2004, The Cryptographers' Track at the RSA Conference 2004, San Francisco, CA, USA, February 23-27, 2004, Proceedings, Lecture Notes in Computer Science, vol. 2964, Springer, 2004, pp. 126–128.

[BP96] Eric Bach and René Peralta, *Asymptotic semismoothness probabilities*, Math. Comp. **65** (1996), no. 216, 1701–1715.

[BP97] Niko Baric and Birgit Pfitzmann, *Collision-free accumulators and fail-stop signature schemes without trees*, Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding, Lecture Notes in Computer Science, vol. 1233, Springer, 1997, pp. 480–494.

[BR93] Mihir Bellare and Phillip Rogaway, *Entity authentication and key distribution*, Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings, Lecture Notes in Computer Science, vol. 773, Springer, 1993, pp. 232–249.

[Bre00] Richard P. Brent, *Public key cryptography with a group of unknown order*, Tech. report, Oxford University, 2000.

[BS07] William D. Banks and Igor E. Shparlinski, *Integers with a large smooth divisor*, Integers **7** (2007), A17, 11.

[Buc90] Johannes Buchmann, *A subexponential algorithm for the determination of class groups and regulators of algebraic number fields*, Séminaire de Théorie des Nombres, Paris 1988–1989, Progr. Math., vol. 91, Birkhäuser Boston, Boston, MA, 1990, pp. 27–41.

[BW88] Johannes Buchmann and Hugh C. Williams, *A key-exchange system based on imaginary quadratic fields*, J. Cryptol. **1** (1988), no. 2, 107–118.

[Can87] David G. Cantor, *Computing in the Jacobian of a hyperelliptic curve*, Math. Comp. **48** (1987), no. 177, 95–101.

[Can94] ———, *On the analogue of the division polynomials for hyperelliptic curves*, J. Reine Angew. Math. **447** (1994), 91–145.

[CCD$^+$20] Katriel Cohn-Gordon, Cas Cremers, Benjamin Dowling, Luke Garratt, and Douglas Stebila, *A formal security analysis of the Signal messaging protocol*, J. Cryptol. **33** (2020), no. 4, 1914–1983.

[CCL$^+$19] Guilhem Castagnos, Dario Catalano, Fabien Laguillaumie, Federico Savasta, and Ida Tucker, *Two-party ECDSA from hash proof systems and efficient instantiations*, Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part III, Lecture Notes in Computer Science, vol. 11694, Springer, 2019, pp. 191–221.

[CCL$^+$21] ———, *Bandwidth-efficient threshold EC-DSA revisited: Online/offline extensions, identifiable aborts, proactivity and adaptive security*, Cryptology ePrint Archive, Report 2021/291, 2021, `https://ia.cr/2021/291`.

[CFA$^+$12] Henri Cohen, Gerhard Frey, Roberto Avanzi, Christophe Doche, Tanja Lange, Kim Nguyen, and Frederik Vercauteren, *Handbook of elliptic and hyperelliptic curve cryptography*, 2nd ed., Chapman & Hall/CRC, 2012.

[CJ88] George E. Collins and Jeremy R. Johnson, *The probability of relative primality of Gaussian integers*, Symbolic and Algebraic Computation, International Symposium ISSAC'88, Rome, Italy, July 4-8, 1988, Proceedings, Lecture Notes in Computer Science, vol. 358, Springer, 1988, pp. 252–258.

[CK01] Ran Canetti and Hugo Krawczyk, *Analysis of key-exchange protocols and their use for building secure channels*, Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding, Lecture Notes in Computer Science, vol. 2045, Springer, 2001, pp. 453–474.

[CL84] Henri Cohen and Hendrik W. Lenstra, Jr., *Heuristics on class groups of number fields*, Number theory, Noordwijkerhout 1983, Lecture Notes in Math., vol. 1068, Springer, Berlin, 1984, pp. 33–62.

[CL02] Jan Camenisch and Anna Lysyanskaya, *Dynamic accumulators and application to efficient revocation of anonymous credentials*, Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings, Lecture Notes in Computer Science, vol. 2442, Springer, 2002, pp. 61–76.

[CL11] Craig Costello and Kristin E. Lauter, *Group law computations on Jacobians of hyperelliptic curves*, Selected Areas in Cryptography - 18th International Workshop, SAC 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers, Lecture Notes in Computer Science, vol. 7118, Springer, 2011, pp. 92–117.

[CLG09] Denis X. Charles, Kristin E. Lauter, and Eyal Z. Goren, *Cryptographic hash functions from expander graphs*, J. Cryptology **22** (2009), no. 1, 93–113.

[CLM$^+$18] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes, *CSIDH: An efficient post-quantum commutative group action*, Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part III, Lecture Notes in Computer Science, vol. 11274, Springer, 2018, pp. 395–427.

[CLN16] Craig Costello, Patrick Longa, and Michael Naehrig, *Efficient algorithms for supersingular isogeny Diffie–Hellman*, Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I, Lecture Notes in Computer Science, vol. 9814, Springer, 2016, pp. 572–601.

[Coh10] Henri Cohen, *A course in computational algebraic number theory*, Graduate Texts in Mathematics, vol. 138, Springer, 2010.

[COR21] Daniel Cervantes-Vázquez, Eduardo Ochoa-Jiménez, and Francisco Rodríguez-Henríquez, *Extended supersingular isogeny Diffie–Hellman key exchange protocol: Revenge of the SIDH*, IET Information Security (2021).

[Cou06] Jean-Marc Couveignes, *Hard homogeneous spaces*, Cryptology ePrint Archive, Report 2006/291, 2006, `https://ia.cr/2006/291`.

[Cox89] David A. Cox, *Primes of the form $x^2 + ny^2$: Fermat, class field theory, and complex multiplication*, Monographs and textbooks in pure and applied mathematics, Wiley, 1989.

[Cre09] Cas J. F. Cremers, *Formally and practically relating the CK, CK-HMQV, and eCK security models for authenticated key exchange*, Cryptology ePrint Archive, Report 2009/253, 2009, `https://ia.cr/2009/253`.

[CY21] Handong Cui and Tsz Hon Yuen, *A trustless GQ multi-signature scheme with identifiable abort*, Information Security and Privacy - 26th Australasian Conference, ACISP 2021, Virtual Event, December 1-3, 2021, Proceedings, Lecture Notes in Computer Science, vol. 13083, Springer, 2021, pp. 673–693.

[DCP$^+$19] Jintai Ding, Ming-Shing Chen, Albrecht Petzoldt, Dieter Schmidt, and Bo-Yin Yang, *Rainbow*, Tech. report, 2019, Available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions`.

[DDGZ21] Luca De Feo, Samuel Dobson, Steven D. Galbraith, and Lukas Zobernig, *SIDH* *proof of knowledge*, Cryptology ePrint Archive, Report 2021/1023, 2021, `https://ia.cr/2021/1023`.

[DG19] Samuel Dobson and Steven D. Galbraith, *On the degree-insensitive SI-GDH problem and assumption*, Cryptology ePrint Archive, Report 2019/929, 2019, `https://ia.cr/2019/929`.

[DG21] ———, *Post-quantum Signal key agreement with SIDH*, Cryptology ePrint Archive, Report 2021/1187, 2021, `https://ia.cr/2021/1187`.

[DGL$^+$20] Samuel Dobson, Steven D. Galbraith, Jason T. LeGrow, Yan Bo Ti, and Lukas Zobernig, *An adaptive attack on 2-SIDH*, Int. J. Comput. Math. Comput. Syst. Theory **5** (2020), no. 4, 282–299.

[DGS21] Samuel Dobson, Steven D. Galbraith, and Benjamin Smith, *Trustless unknown-order groups*, presented at MathCrypt, 2021, `https://ia.cr/2020/196`.

[DH76] Whitfield Diffie and Martin E. Hellman, *New directions in cryptography*, IEEE Trans. Inf. Theory **22** (1976), no. 6, 644–654.

[Die06] Claus Diem, *An index calculus algorithm for plane curves of small degree*, Algorithmic number theory, Lecture Notes in Comput. Sci., vol. 4076, Springer, Berlin, 2006, pp. 543–557.

[DJP14] Luca De Feo, David Jao, and Jérôme Plût, *Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies*, J. Math. Cryptol. **8** (2014), no. 3, 209–247.

[DKL$^+$18] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé, *CRYSTALS-Dilithium: A lattice-based digital signature scheme*, IACR Trans. Cryptogr. Hardw. Embed. Syst. **2018** (2018), no. 1, 238–268.

[DKL$^+$20] Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski, *SQISign: Compact post-quantum signatures from quaternions and isogenies*, Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part I, Lecture Notes in Computer Science, vol. 12491, Springer, 2020, pp. 64–93.

[Dry19] Thaddeus Dryja, *Utreexo: A dynamic hash-based accumulator optimized for the Bitcoin UTXO set*, Cryptology ePrint Archive, Report 2019/611, 2019, `https://ia.cr/2019/611`.

[Dui19] Ines Duits, *The post-quantum Signal protocol: Secure chat in a quantum world*, Master's thesis, University of Twente, 2019.

[Eng02] Andreas Enge, *Computing discrete logarithms in high-genus hyperelliptic Jacobians in provably subexponential time*, Math. Comp. **71** (2002), no. 238, 729–742.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto, *Secure integration of asymmetric and symmetric encryption schemes*, Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings, Lecture Notes in Computer Science, vol. 1666, Springer, 1999, pp. 537–554.

[FO13] ———, *Secure integration of asymmetric and symmetric encryption schemes*, J. Cryptol. **26** (2013), no. 1, 80–101.

[FP21] Tako Boris Fouotsa and Christophe Petit, *SHealS and HealS: Isogeny-based PKEs from a key validation method for SIDH*, Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part IV, Lecture Notes in Computer Science, vol. 13093, Springer, 2021, pp. 279–307.

[FP22] ———, *A new adaptive attack on SIDH*, Topics in Cryptology - CT-RSA 2022 - Cryptographers' Track at the RSA Conference 2022, Virtual Event, March 1-2, 2022, Proceedings, Lecture Notes in Computer Science, vol. 13161, Springer, 2022, pp. 322–344.

[FR94] Gerhard Frey and Hans-Georg Rück, *A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves*, Math. Comp. **62** (1994), no. 206, 865–874.

[FSXY12] Atsushi Fujioka, Koutarou Suzuki, Keita Xagawa, and Kazuki Yoneyama, *Strongly secure authenticated key exchange from factoring, codes, and lattices*, Public Key Cryptography - PKC 2012 - 15th International Conference on Practice and Theory in Public Key Cryptography, Darmstadt, Germany, May 21-23, 2012. Proceedings, Lecture Notes in Computer Science, vol. 7293, Springer, 2012, pp. 467–484.

[FTTY18] Atsushi Fujioka, Katsuyuki Takashima, Shintaro Terada, and Kazuki Yoneyama, *Supersingular isogeny Diffie–Hellman authenticated key exchange*, Information Security and Cryptology - ICISC 2018 - 21st International Conference, Seoul, South Korea, November 28-30, 2018, Revised Selected Papers, Lecture Notes in Computer Science, vol. 11396, Springer, 2018, pp. 177–195.

[FWG07] Xinxin Fan, Thomas J. Wollinger, and Guang Gong, *Efficient explicit formulae for genus 3 hyperelliptic curve cryptosystems over binary fields*, IET Inf. Secur. **1** (2007), no. 2, 65–81.

[Gal12] Steven D. Galbraith, *Mathematics of public key cryptography*, Cambridge University Press, Cambridge, 2012.

[Gal18] ———, *Authenticated key exchange for SIDH*, Cryptology ePrint Archive, Report 2018/266, 2018, `https://ia.cr/2018/266`.

[Gau66] Carl Friedrich Gauss, *Disquisitiones arithmeticae*, translated into English by Arthur A. Clarke, Yale University Press, 1966, original text in Latin, 1801.

[Gau00] Pierrick Gaudry, *An algorithm for solving the discrete log problem on hyperelliptic curves*, Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding, Lecture Notes in Computer Science, vol. 1807, Springer, 2000, pp. 19–34.

[GH00] Pierrick Gaudry and Robert Harley, *Counting points on hyperelliptic curves over finite fields*, Algorithmic number theory (Leiden, 2000), Lecture Notes in Comput. Sci., vol. 1838, Springer, Berlin, 2000, pp. 313–332.

[GHV07] Steven D. Galbraith, Florian Hess, and Frederik Vercauteren, *Hyperelliptic pairings*, Pairing-Based Cryptography - Pairing 2007, First International Conference, Tokyo, Japan, July 2-4, 2007, Proceedings, Lecture Notes in Computer Science, vol. 4575, Springer, 2007, pp. 108–131.

[GJ79] Michael R. Garey and David S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness*, A Series of Books in the Mathematical Sciences, W. H. Freeman and Co., San Francisco, Calif., 1979.

[GKS11] Pierrick Gaudry, David Kohel, and Benjamin Smith, *Counting points on genus 2 curves with real multiplication*, Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings, Lecture Notes in Computer Science, vol. 7073, Springer, 2011, pp. 504–519.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson, *Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems*, J. ACM **38** (1991), no. 3, 691–729.

[GPS20] Steven D. Galbraith, Christophe Petit, and Javier Silva, *Identification protocols and signature schemes based on supersingular isogeny problems*, J. Cryptol. **33** (2020), no. 1, 130–175.

[GPST16] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti, *On the security of supersingular isogeny cryptosystems*, Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I, 2016, pp. 63–91.

[GPV21] Wissam Ghantous, Federico Pintore, and Mattia Veroni, *Collisions in supersingular isogeny graphs and the SIDH-based identification protocol*, Cryptology ePrint Archive, Report 2021/1051, 2021, `https://ia.cr/2021/1051`.

[GS04a] Pierrick Gaudry and Éric Schost, *Construction of secure random curves of genus 2 over prime fields*, Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings, Lecture Notes in Computer Science, vol. 3027, Springer, 2004, pp. 239–256.

[GS04b] ———, *A low-memory parallel version of Matsuo, Chao, and Tsujii's algorithm*, Algorithmic number theory, Lecture Notes in Comput. Sci., vol. 3076, Springer, Berlin, 2004, pp. 208–222.

[GS12] ———, *Genus 2 point counting over prime fields*, J. Symbolic Comput. **47** (2012), no. 4, 368–400.

[GTTD07] Pierrick Gaudry, Emmanuel Thomé, Nicolas Thériault, and Claus Diem, *A double large prime variation for small genus hyperelliptic index calculus*, Math. Comput. **76** (2007), no. 257, 475–492.

[GV18] Steven D. Galbraith and Frederik Vercauteren, *Computational problems in supersingular elliptic curve isogenies*, Quantum Inf. Process. **17** (2018), no. 10, 265.

[Har07] David Harvey, *Kedlaya's algorithm in larger characteristic*, Int. Math. Res. Not. IMRN (2007), no. 22, Art. ID rnm095, 29.

[HHK17] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz, *A modular analysis of the Fujisaki–Okamoto transformation*, Theory of Cryptography - 15th International

Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part I, Lecture Notes in Computer Science, vol. 10677, Springer, 2017, pp. 341–371.

[HKKP21] Keitaro Hashimoto, Shuichi Katsumata, Kris Kwiatkowski, and Thomas Prest, *An efficient and generic construction for Signal's handshake (X3DH): Post-quantum, state leakage secure, and deniable*, Public-Key Cryptography - PKC 2021 - 24th IACR International Conference on Practice and Theory of Public Key Cryptography, Virtual Event, May 10-13, 2021, Proceedings, Part II, Lecture Notes in Computer Science, vol. 12711, Springer, 2021, pp. 410–440.

[HL10] Carmit Hazay and Yehuda Lindell, *Sigma protocols and efficient zero-knowledge*, pp. 147–175, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[HM89] James L. Hafner and Kevin S. McCurley, *A rigorous subexponential algorithm for computation of class groups*, J. Amer. Math. Soc. **2** (1989), no. 4, 837–850.

[HM00] Safuat Hamdy and Bodo Möller, *Security of cryptosystems based on class groups of imaginary quadratic orders*, Advances in Cryptology - ASIACRYPT 2000, 6th International Conference on the Theory and Application of Cryptology and Information Security, Kyoto, Japan, December 3-7, 2000, Proceedings, Lecture Notes in Computer Science, vol. 1976, Springer, 2000, pp. 234–247.

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman, *NTRU: A ring-based public key cryptosystem*, Algorithmic Number Theory, Third International Symposium, ANTS-III, Portland, Oregon, USA, June 21-25, 1998, Proceedings, Lecture Notes in Computer Science, vol. 1423, Springer, 1998, pp. 267–288.

[HSS01] Florian Hess, Gadiel Seroussi, and Nigel P. Smart, *Two topics in hyperelliptic cryptography*, Selected Areas in Cryptography, 8th Annual International Workshop, SAC 2001 Toronto, Ontario, Canada, August 16-17, 2001, Revised Papers, Lecture Notes in Computer Science, vol. 2259, Springer, 2001, pp. 181–189.

[IBM21] IBM, *IBM Quantum breaks the 100-qubit processor barrier*, `https://research.ibm.com/blog/127-qubit-quantum-processor-eagle`, 2021, Accessed: 2022-01-18.

[JD11] David Jao and Luca De Feo, *Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies*, Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29 - December 2, 2011. Proceedings, Lecture Notes in Computer Science, vol. 7071, Springer, 2011, pp. 19–34.

[JKL04] Ik Rae Jeong, Jonathan Katz, and Dong Hoon Lee, *One-round protocols for two-party authenticated key exchange*, Applied Cryptography and Network Security, Second International Conference, ACNS 2004, Yellow Mountain, China, June 8-11, 2004, Proceedings, Lecture Notes in Computer Science, vol. 3089, Springer, 2004, pp. 220–232.

[JS14] David Jao and Vladimir Soukharev, *Isogeny-based quantum-resistant undeniable signatures*, Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014, Waterloo, ON, Canada, October 1-3, 2014. Proceedings, Lecture Notes in Computer Science, vol. 8772, Springer, 2014, pp. 160–179.

[Ked01] Kiran S. Kedlaya, *Counting points on hyperelliptic curves using Monsky–Washnitzer cohomology*, J. Ramanujan Math. Soc. **16** (2001), no. 4, 323–338.

[KLM$^+$15] Daniel Kirkwood, Bradley C. Lackey, John McVey, Mark Motley, Jerome A. Solinas, and David Tuller, *Failure is not an option: Standardization issues for post-quantum key agreement*, 2015, Workshop on Cybersecurity in a Post-Quantum World.

[Kob89] Neal Koblitz, *Hyperelliptic cryptosystems*, J. Cryptol. **1** (1989), no. 3, 139–150.

[KPG99] Aviad Kipnis, Jacques Patarin, and Louis Goubin, *Unbalanced oil and vinegar signature schemes*, Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding, Lecture Notes in Computer Science, vol. 1592, Springer, 1999, pp. 206–222.

[Kra05] Hugo Krawczyk, *HMQV: A high-performance secure Diffie–Hellman protocol*, Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings, Lecture Notes in Computer Science, vol. 3621, Springer, 2005, pp. 546–566.

[Lag75] Joseph-Louis Lagrange, *Recherches d'arithmétique*, C.F. Voss, 1775.

[Lai15] Kim Laine, *Security of genus 3 curves in cryptography*, Ph.D. thesis, University of California, Berkeley, 2015.

[Lam79] Leslie Lamport, *Constructing digital signatures from a one way function*, Tech. Report CSL-98, October 1979, This paper was published by IEEE in the Proceedings of HICSS-43 in January, 2010.

[Lan05] Tanja Lange, *Formulae for arithmetic on genus 2 hyperelliptic curves*, Appl. Algebra Eng. Commun. Comput. **15** (2005), no. 5, 295–328.

[Lee20] Jonathan Lee, *The security of groups of unknown order based on Jacobians of hyperelliptic curves*, Cryptology ePrint Archive, Report 2020/289, 2020, `https://ia.cr/2020/289`.

[Leo20] Christopher Leonardi, *A note on the ending elliptic curve in SIDH*, Cryptology ePrint Archive, Report 2020/262, 2020, `https://ia.cr/2020/262`.

[Lip12] Helger Lipmaa, *Secure accumulators from Euclidean rings without trusted setup*, Applied Cryptography and Network Security - 10th International Conference, ACNS 2012, Singapore, June 26-29, 2012. Proceedings, Lecture Notes in Computer Science, vol. 7341, Springer, 2012, pp. 224–240.

[LL15] Kim Laine and Kristin E. Lauter, *Time-memory trade-offs for index calculus in genus 3*, J. Math. Cryptol. **9** (2015), no. 2, 95–114.

[LLM07] Brian A. LaMacchia, Kristin E. Lauter, and Anton Mityagin, *Stronger security of authenticated key exchange*, Provable Security, First International Conference, ProvSec 2007, Wollongong, Australia, November 1-2, 2007, Proceedings, Lecture Notes in Computer Science, vol. 4784, Springer, 2007, pp. 1–16.

[LLX07] Jiangtao Li, Ninghui Li, and Rui Xue, *Universal accumulators with efficient nonmembership proofs*, Applied Cryptography and Network Security, 5th International Conference, ACNS 2007, Zhuhai, China, June 5-8, 2007, Proceedings (Jonathan Katz and Moti Yung, eds.), Lecture Notes in Computer Science, vol. 4521, Springer, 2007, pp. 253–269.

[LZ19] Qipeng Liu and Mark Zhandry, *Revisiting post-quantum Fiat–Shamir*, Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II, Lecture Notes in Computer Science, vol. 11693, Springer, 2019, pp. 326–355.

[McE78] Robert J. McEliece, *A public-key cryptosystem based on algebraic coding theory*, Deep Space Network Progress Report **44** (1978), 114–116.

[Mer79] Ralph Merkle, *Secrecy, authentication and public key systems: A certified digital signature*, Thesis, Dept. of Electrical Engineering, Stanford University, 1979.

[MI88] Tsutomu Matsumoto and Hideki Imai, *Public quadratic polynominal-tuples for efficient signature-verification and message-encryption*, Advances in Cryptology - EUROCRYPT '88, Workshop on the Theory and Application of of Cryptographic Techniques, Davos, Switzerland, May 25-27, 1988, Proceedings, Lecture Notes in Computer Science, vol. 330, Springer, 1988, pp. 419–453.

[MP16a] Moxie Marlinspike and Trevor Perrin, *The double ratchet algorithm*, `https://signal.org/docs/specifications/doubleratchet/`, 2016, Revision 1, 2016-11-20.

[MP16b] ———, *The X3DH key agreement protocol*, `https://signal.org/docs/specifications/x3dh/`, 2016, Revision 1, 2016-11-04.

[MP19] Chloe Martindale and Lorenz Panny, *How to not break SIDH*, CFAIL, 2019.

[Mum07] David Mumford, *Tata lectures on theta II*, Birkhäuser, 01 2007.

[Mum08] ———, *Abelian varieties*, Tata Institute of Fundamental Research Studies in Mathematics, vol. 5, Published for the Tata Institute of Fundamental Research, Bombay; by Hindustan Book Agency, New Delhi, 2008, With appendices by C. P. Ramanujam and Yuri Manin, Corrected reprint of the second (1974) edition.

[MWZ96] Alfred Menezes, Yi-Hong Wu, and Robert J. Zuccherato, *An elementary introduction to hyperelliptic curves*, Appendix in Algebraic Aspects of Cryptography by Neal Koblitz, Springer-Verlag, 1998, pages 155-178, 1996.

[Nag04] Koh-ichi Nagao, *Improvement of Thériault algorithm of index calculus for Jacobian of hyperelliptic curves of small genus*, Cryptology ePrint Archive, Report 2004/161, 2004, `https://ia.cr/2004/161`.

[Pat95] Jacques Patarin, *Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt'88*, Advances in Cryptology - CRYPTO '95, 15th Annual International Cryptology Conference, Santa Barbara, California, USA, August 27-31, 1995, Proceedings, Lecture Notes in Computer Science, vol. 963, Springer, 1995, pp. 248–261.

[Pei14] Chris Peikert, *Lattice cryptography for the internet*, Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014, Waterloo, ON, Canada, October 1-3, 2014. Proceedings, Lecture Notes in Computer Science, vol. 8772, Springer, 2014, pp. 197–219.

[Per16] Trevor Perrin, *The XEdDSA and VXEdDSA signature schemes*, `https://signal.org/docs/specifications/xeddsa/`, 2016, Revision 1, 2016-10-20.

[Pil90] Jonathan Pila, *Frobenius maps of abelian varieties and finding roots of unity in finite fields*, Math. Comp. **55** (1990), no. 192, 745–763.

[Pit09] Nicole L. Pitcher, *Efficient point-counting on genus-2 hyperelliptic curves*, Ph.D. thesis, University of Illinois at Chicago, 2009, p. 124.

[Piz98] Arnold K. Pizer, *Ramanujan graphs*, Computational perspectives on number theory (Chicago, IL, 1995), AMS/IP Stud. Adv. Math., vol. 7, Amer. Math. Soc., Providence, RI, 1998, pp. 159–178.

[RS06] Alexander Rostovtsev and Anton Stolbunov, *Public-key cryptosystem based on isogenies*, Cryptology ePrint Archive, Report 2006/145, 2006, `https://ia.cr/2006/145`.

[RSW96] Ronald L. Rivest, Adi Shamir, and David A. Wagner, *Time-lock puzzles and timed-release crypto*, Technical Report MIT/LCS/TR-684, 1996.

[Rüc99] Hans-Georg Rück, *On the discrete logarithm in the divisor class group of curves*, Math. Comp. **68** (1999), no. 226, 805–806.

[San99] Tomas Sander, *Efficient accumulators without trapdoor (extended abstract)*, Information and Communication Security, Second International Conference, ICICS'99, Sydney, Australia, November 9-11, 1999, Proceedings, Lecture Notes in Computer Science, vol. 1726, Springer, 1999, pp. 252–262.

[Sch85] René Schoof, *Elliptic curves over finite fields and the computation of square roots mod p*, Math. Comp. **44** (1985), no. 170, 483–494.

[Sch91] Claus-Peter Schnorr, *Efficient signature generation by smart cards*, J. Cryptol. **4** (1991), no. 3, 161–174.

[Sch95] René Schoof, *Counting points on elliptic curves over finite fields*, J. Théor. Nombres Bordeaux **7** (1995), no. 1, 219–254, Les Dix-huitièmes Journées Arithmétiques (Bordeaux, 1993).

[Sho94] Peter W. Shor, *Algorithms for quantum computation: Discrete logarithms and factoring*, 35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994, IEEE Computer Society, 1994, pp. 124–134.

[Sho97] ______, *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*, SIAM J. Comput. **26** (1997), no. 5, 1484–1509.

[Sil09] Joseph H. Silverman, *The arithmetic of elliptic curves*, 2nd ed., Graduate Texts in Mathematics, vol. 106, Springer, Dordrecht, 2009.

[Smi09] Benjamin Smith, *Isogenies and the discrete logarithm problem in Jacobians of genus 3 hyperelliptic curves*, J. Cryptol. **22** (2009), no. 4, 505–529.

[SSW20] Peter Schwabe, Douglas Stebila, and Thom Wiggers, *Post-quantum TLS without handshake signatures*, CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020, ACM, 2020, pp. 1461–1480.

[Sut07] Andrew V. Sutherland, *Order computations in generic groups*, Ph.D. thesis, Massachusetts Institute of Technology, 2007.

[Sut09] ______, *A generic approach to searching for Jacobians*, Math. Comput. **78** (2009), no. 265, 485–507.

[Tat66] John Tate, *Endomorphisms of abelian varieties over finite fields*, Invent. Math. **2** (1966), 134–144.

[Tes99] Edlyn Teske, *The Pohlig–Hellman method generalized for group structure computation*, J. Symb. Comput. **27** (1999), no. 6, 521–534.

[Tha20] Steve Thakur, *Constructing hidden order groups using genus three Jacobians*, Cryptology ePrint Archive, Report 2020/348, 2020, `https://ia.cr/2020/348`.

[Thé03] Nicolas Thériault, *Index calculus attack for hyperelliptic curves of small genus*, Advances in Cryptology - ASIACRYPT 2003, 9th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, November 30 - December 4, 2003, Proceedings, Lecture Notes in Computer Science, vol. 2894, Springer, 2003, pp. 75–92.

[Tho17] Erik Thormarker, *Post-quantum cryptography: Supersingular isogeny Diffie–Hellman key exchange*, Thesis, Stockholm University, 2017.

[UG18] Nik Unger and Ian Goldberg, *Improved strongly deniable authenticated key exchanges for secure messaging*, Proc. Priv. Enhancing Technol. **2018** (2018), no. 1, 21–66.

[UJ18] David Urbanik and David Jao, *SoK: The problem landscape of SIDH*, Proceedings of the 5th ACM on ASIA Public-Key Cryptography Workshop, APKC AsiaCCS, Incheon, Republic of Korea, June 4, 2018, ACM, 2018, pp. 53–60.

[UJ20] ———, *New techniques for SIDH-based NIKE*, J. Math. Cryptol. **14** (2020), no. 1, 120–128.

[UXT$^+$22] Rei Ueno, Keita Xagawa, Yutaro Tanaka, Akira Ito, Junko Takahashi, and Naofumi Homma, *Curse of re-encryption: A generic power/EM analysis on post-quantum KEMs*, IACR Transactions on Cryptographic Hardware and Embedded Systems (2022), 296–322.

[Vél71] Jacques Vélu, *Isogénies entre courbes elliptiques*, C. R. Acad. Sci. Paris Sér. A-B **273** (1971), A238–A241.

[VGIK20] Nihal Vatandas, Rosario Gennaro, Bertrand Ithurburn, and Hugo Krawczyk, *On the cryptographic deniability of the Signal protocol*, Applied Cryptography and Network Security - 18th International Conference, ACNS 2020, Rome, Italy, October 19-22, 2020, Proceedings, Part II, Lecture Notes in Computer Science, vol. 12147, Springer, 2020, pp. 188–209.

[Wen01] Annegret Weng, *A class of hyperelliptic CM-curves of genus three*, J. Ramanujan Math. Soc. **16** (2001), no. 4, 339–372.

[Wen03] ———, *Constructing hyperelliptic curves of genus 2 suitable for cryptography*, Math. Comp. **72** (2003), no. 241, 435–458.

[Wes19] Benjamin Wesolowski, *Efficient verifiable delay functions*, Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part III, Lecture Notes in Computer Science, vol. 11478, Springer, 2019, pp. 379–407.

[YAJ$^+$17] Youngho Yoo, Reza Azarderakhsh, Amir Jalali, David Jao, and Vladimir Soukharev, *A post-quantum digital signature scheme based on supersingular isogenies*, Financial Cryptography and Data Security - 21st International Conference, FC 2017, Sliema, Malta, April 3-7, 2017, Revised Selected Papers, 2017, pp. 163–181.

[Zag81] Don Zagier, *Zetafunktionen und quadratische körper: Eine einführung in die höhere zahlentheorie*, Hochschultext (Berlin), Springer-Verlag, 1981.