Cryptanalysis of Isogeny-based cryptosystems in a leakage model

Subham Das

A dissertation submitted for the partial fulfilment of BS-MS dual degree in Science

Indian Institute of Science Education and Research, Mohali

April, 2025

Certificate of Examination

This is to certify that the dissertation titled **Cryptanalysis of Isogeny-based cryptosystems in a leakage model** submitted by **Subham Das** (Reg. No. MS20121) for the partial fulfillment of BS- MS Dual Degree programme of the institute, has been examined by the thesis committee duly appointed by the institute. The committee finds the work done by the candidate satisfactory and recommends that the report be accepted.

Prof. Vaibhav Vaish | Prof. Shane D'Mello | Prof. Abhik Ganguli

Prof. Abhik Ganguli
(Local Supervisor)

Prof. Péter Kutas
(External Supervisor)

Dated: .

Declaration

The work presented in this dissertation has been carried out by me under the guidance of **Prof. Péter Kutas** from **Eötvös Loràand University** and **Prof. Abhik Ganguli** at the Indian Institute of Science Education and Research, Mohali.

This work has not been submitted in part or in full for a degree, a diploma, or a fellowship to any other university or institute. Whenever contributions of others are involved, every effort is made to indicate this clearly, with due acknowledgement of collaborative research and discussions. This thesis is a bonafide record of original work done by me and all sources listed within have been detailed in the bibliography.

Subham Das
(Candidate)
Dated: .

In my capacity as the supervisor of the candidate project work, I certify that the above statements by the candidate are true to the best of my knowledge.

Prof. Abhik Ganguli
(Local Supervisor)

Prof. Péter Kutas
(External Supervisor)
Dated: .

Acknowledgements

I would like to begin by expressing my gratitude to Prof. Péter Kutas's thorough guidance, insights, his constant encouragement and support throughout the duration of the project has made it possible for me to compose this thesis. My sincere thanks for showing me the beauty of cryptography and the opportunity to work on this project. I would like to thank Prof. Abhik Ganguli, for his supervision, encouragement, feedback and seeing through the completion of the thesis. I shall be ever grateful for your insightful advice about the varied aspects of math, research and academia every time we have interacted. I am grateful to my thesis committee members, Prof. Shane D'Mello and Prof. Vaibhav Vaish, for their efforts to provide constructive feedback and evaluate my Master's thesis. I also extend my heartfelt thanks to my research collaborators, Riccardo and Jonas, for the opportunity to discuss and proliferate ideas, which indeed has enriched my understanding of the subject.

I would like to thank IISER Mohali for giving me the opportunity to undertake this research as a part of my Master's thesis project. I would like to thank the Faculty of Informatics, ELTE for hosting my research visit to Prof. Péter Kutas. I would like to acknowledge the Tempus Public Foundation for awarding me the Stipendium Hungaricum scholarship, which facilitated my research visit to Eötvös Loránd University.

I would like to thank and express gratitude to my parents: my late father, to whom I dedicate this thesis to, without whose blood and sweat I would not be in this position today, and without whose care and love I would not be able to grow as a human being, and my mother, whose patience, love and endurance through the past years has sustained me, without whose persistent support and encouragement neither this work, nor the completion of my degree would be possible.

I thank Rabsan, Soham,Sahil and Nele for their constant support throughout the duration of the thesis undertaking, in innumerable ways both academic and otherwise. I thank Shreyas, Manvendra, Rajdeep and Joshua for exploring with me the nooks and crannies of math and academia, and for tolerating me as a fellow math major.

Finally, in all humility do I acknowledge that this work neither depends on my will nor my exertion, but on God, who has mercy.

Abstract

In this thesis we shall investigate and analyse three isogeny-based cryptosystems: M-SIDH, FESTA and POKÉ, all of which are proposed countermeasures to the attacks against SIDH. We analyse them in the 'leakage model' where the core assumption is that partial information about the cryptosystems' internal secret state becomes available to the adversary. We present novel attacks in this scenario, against the cryptosystems and fill the absence of such analysis in contemporary research literature on isogeny-based cryptosystems.

সারাংশ

এই নিবন্ধতে আমরা তদন্ত এবং বিশ্লেষণ করবো তিনটি আইসোজেনি-ভিত্তিক তথ্যগুপ্তিতন্ত্র: এম-সাইঢ (M-SIDH), ফেসটা (FESTA) এবং পোকে (POKÉ), যেগুলি সাইঢ (SIDH) তথ্যগুপ্তিতন্ত্রর বিরুদ্ধে আক্রমণ নিরোধের প্রকল্প | আমরা এই তথ্যগুপ্তিতন্ত্র-গুলিকে একটি 'ছিদ্রগুন আদর্শে ' বিশ্লেষণ করবো, যেই আদর্শের মূল অনুমান হলো যে তথ্যগুপ্তিতন্ত্রের বিরুদ্ধে কোনো প্রতিপক্ষ সেই তন্ত্রের অভ্যন্তরীণ অবস্থার ব্যাপারে কিছু আংশিক তথ্য লাভ করতে পারে এবং সেটি যন্ত্রটার বিরুদ্ধে আক্রমণে ব্যবহার করতে পারে | এই নিবন্ধে, উপরে বর্ণিত প্রেক্ষাপটে আমরা এই তিনটি তথ্যগুপ্তিতন্ত্রের বিরুদ্ধে কিছু অভিনব আক্রমণ ব্যাখ্যা করবো এবং আইসোজেনি-ভিত্তিক তথ্যগুপ্তিতন্ত্র জড়িত বর্তমান গবেষণা সাহিত্যে এইরকম বিশ্লেষণের অভাব পূর্ণ করবো |

List of Algorithms

1 Diffie-Hellman protocol . . . 3
2 LLL algorithm (with Euclidean norm and $\delta = 3/4$) . . . 17
3 Buchberger's algorithm . . . 19
4 Coppersmith's method . . . 26
5 Constructing an optimal set $\mathcal{F}$ . . . 28
6 $\mathsf{SIDH.isogen}_i$ computing public keys . . . 30
7 $\mathsf{SIDH.isogex}_i$ shared key establishment . . . 31
8 $\mathsf{SIDH.Enc}_i$ encryption algorithm . . . 31
9 $\mathsf{SIDH.Dec}_i$ decryption algorithm . . . 31
10 $f(m = (\langle K_1 \rangle, \langle K_2 \rangle, \Gamma'), \mathsf{pk})$ trapdoor function . . . 39
11 $f^{-1}(m = (E_1, R_1, S_1, E_2, R_2, S_2)$ trapdoor inversion algorithm . . . 40
12 $\mathsf{FESTA.Enc}(\mathsf{pk}, m)$ encryption algorithm . . . 41
13 $\mathsf{FESTA.Dec}(\mathsf{sk}, ct)$ decryption method . . . 42
14 Generating a $q(2^a - q)$-isogeny . . . 45
15 POKE.Enc encryption algorithm . . . 46
16 POKE.Dec decryption algorithm . . . 47

Contents

Abstract VI

সারাংশ VII

List of Algorithms VIII

1 Introduction 2

2 Preliminaries 7

2.1 Elliptic Curves and Isogenies . . . 7

2.1.1 Elliptic curves . . . 7

2.1.2 Elliptic curves over finite fields . . . 11

2.1.3 Pairings and Isogeny Representations . . . 11

2.2 Lattice theory . . . 15

2.2.1 Basic notions . . . 15

2.2.2 Lattice reduction . . . 16

2.3 Gröbner Bases . . . 18

3 Finding small roots of polynomials 20

3.1 Coppersmith's method . . . 20

3.1.1 Univariate case . . . 21

3.1.2 Bivariate case . . . 22

3.1.3 Multivariate case . . . 23

3.2 Automated Coppersmith . . . 24

3.2.1 Procedure . . . 25

3.2.2 Parameter choices . . . 26

4 Isogeny-based cryptosystems 29

4.1 SIDH: Supersingular Isogeny Diffie Hellman . . . 29

4.1.1 Protocol . . . 29

4.1.2 SIDH attack . . . 32

4.2 M-SIDH: Masked Supersingular Isogeny Diffie Hellman 33
4.2.1 Protocol 33
4.2.2 Security Analysis 35
4.2.3 Parameters 37
4.3 FESTA: Fast Encryption from Supersingular Torsion Attacks 38
4.3.1 FESTA trapdoor function 38
4.3.2 Protocol 41
4.3.3 Security Analysis 41
4.3.4 Parameters 42
4.4 POKÉ: POint-based Key Exchange 43
4.4.1 Protocol 44
4.4.2 Security Analysis 48
4.4.3 Parameters 49

5 Cryptanalysis in the bounded-leakage model 50
5.1 Bounded-leakage model 50
5.2 Problem statement 51
5.3 Coppersmith-style attacks 51
5.3.1 M-SIDH 51
5.3.2 FESTA 52
5.3.3 POKÉ 54
5.4 Combinatorial attacks 55
5.4.1 Procedure 55
5.4.2 Complexity 56

6 On the choice of matrices made in FESTA 58
6.1 Introduction 58
6.2 Preliminaries 59
6.2.1 Matrices 59
6.2.2 Level structures 59
6.3 Reduction 59
6.3.1 For $N = p^k$ with odd p 60
6.3.2 For $N = 2^k$ 61
6.3.3 Finite Fields 62

Dedicated to my father, Subhendu Das
(10.01.1957 - 29.06.2022).

Chapter 1

Introduction

Historically, prior to the second half of the 20th century, the discipline of cryptography had mainly existed as an art for constructing and breaking 'ciphers'— an algorithm and an associated secret value known as key, whose objective was to enable secret communication between parties. The cipher allowed one to turn a piece of message incomprehensible to none but the possessor of this key, thereby ensuring the message's confidentiality. This process of breaking and reconstruction would often be guided by intuition and creativity, with little room for development of systematic frameworks, formalization and foundational principles. The establishment of the theory of computation, and the subsequent formalization of algorithms and their complexity in 1940-50's finally paved the way for transformation of cryptography into a rigorous scientific discipline. Today, the field has pervaded deep into society, with applications in banking, public communications, medical services, commerce, journalism etc., besides the age-old use in military and governmental organizations.

The goal of modern cryptography is not to design ciphers (or broadly 'cryptosystems') which rely on perceived complexity or cunning, but designing cryptosystems where the possibility of an adversary learning secret information would be equivalent to expenditure of a large computational resource[1]. A cryptosystem is said to be secure against a particular adversary, when such an equivalence can be shown in a rigorous fashion.[2]

An exemplary feat of modern cryptography is the existence of key-exchange protocols. It is akin to sharing a secret key with your intended partner by first performing a local computation, and then shouting it across a room filled with untrusted people. The invention of such protocols not only radically changed the research-community's

[1] i.e, the computational cost is often larger than any realistic attacker can afford.

[2] There also exists *information-theoretic* security, where the requirement is absolutely zero information is leaked, even to an adversary with unlimited computational power. However for all practical purposes, this condition is unnecessarily strong.

approach to the field, but also ushered cryptography from the private domain to the public domain. The first key-exchange protocol was proposed by Diffie and Hellman [DH76] in 1976, prior to which it was believed that secure communication was impossible without first establishing a secure channel of communication between two parties. The authors observed the existence of an asymmetry, in terms of a procedure which can be easily performed yet extremely hard to reverse,[3] which they used to the advantage that allowed two parties to perform local operations on the message and share it to another via a public channel, such that anyone but the intended receiver cannot reverse the operation.

We now define the Diffie-Hellman key exchange protocol formally, as it appeared in [DH76], in the group theoretic setting. Let $\mathcal{G}$ be a polynomial-time algorithm, which on input of a n-bit string λ outputs a description of the group $\mathbb{G}$, its order q (with $||q|| = n$) and a generator $g \in \mathbb{G}$.

Algorithm 1 Diffie-Hellman protocol

Input: A n-bit string λ.

1: Alice runs $\mathcal{G}(\lambda)$ to obtain $(\mathbb{G}, q, g)$.
2: Alice chooses a uniform $x \in \mathbb{Z}_q$, and computes $h_A := g^x$
3: Alice sends $(\mathbb{G}, q, g, h_A)$ to Bob.
4: Bob receives $(\mathbb{G}, q, g, h_A)$. Bob chooses a uniform $y \in \mathbb{Z}_q$, and computes $h_B := g^y$
5: Bob sends h_B to Alice and outputs the key $k_B := h_A^y = g^{xy}$
6: Alice receives h_B and outputs the key $k_A := h_B^x$

An in depth discussion of the DH protocol and key-exchange protocols can be found in [KL20]. It must be noted that soon after the publication of this protocol, it was widely adapted to various other scenarios due to the flexibility it yields. A notable adaptation of the DH Protocol was the case of elliptic curves, which gave us the Elliptic curve Diffie-hellman (ECDH) protocol. ECDH went on to be adopted as one of the most widely used cryptosystems due to its compactness, efficiency and portability while still maintaining strong security.

Along with the developments in cryptography, the mid-1980s, a new kind of computing system was envisioned which was based on the laws of quantum physics. The classical computer uses bits that can be either 0 or 1 state, whereas the quantum computer uses qubits (analogous to bits) that are either 1 or 0, or in the infinitely many superposition of the two states. At the end of a computation procedure, the

[3] such as given a collection of large primes, it is easy to find its product, but extremely hard to recover the collection if the product is given. This problem is indeed hard for classical computers, but as we shall see it is not the case with quantum ones.

qubit is measured, which forces it to be either 1 or 0, with a certain probability. The properties of superposition and entanglement of quantum states allow quantum computers to solve certain class of problems much more efficiently than classical computers.

In particular, Peter Shor proposed an algorithm for quantum computers[4] in 1999 which would be able to break cryptosystems like RSA and ECDH, by solving the underlying hard problems such as prime factorization and the discrete logarithm problem respectively. If not bad already, the idea of increasing the parameters of these cryptosystems to make them secure against quantum attacks would render the protocols totally unfit for use. Hence, to provide security against such quantum attacks, one would need to desert these cryptosystems entirely, and seek newer methods.

Towards this ailment, the field of post quantum cryptography comes to aid. It aims to supply hard problems, and cryptosystems built upon them, which would stay secure from both quantum and classical attacks. Under this heading, many different flavours of cryptography come in such as lattice-based, code-based, isogeny-based and multivariate cryptography, each of which have their strengths and weaknesses. In this thesis we shall focus on isogeny-based cryptography.

The field had its inception two decades ago through the work of Couveignes [Cou06], making isogeny-based cryptography one of the youngest fields in cryptology. It has proposed several hard problems involving computation of isogenies of elliptic curves, the primary of which is the following:

Problem 1.0.1. *Given two supersingular elliptic curves, find an isogeny between them.*

Till date, the known quantum and classical attempts towards this problem still remain with exponential time complexity, and remains one of the contenders for quantum-hard problems. A number of cryptographic protocols have been proposed in the recent past based on Problem 1.0.1, a notable one being the Supersingular Isogeny Diffie Hellman (SIDH) protocol.

The SIDH protocol, proposed in 2011 [Jao22], was an adaptation of the classical Diffie-Hellman key-exchange protocol to the isogeny framework, which was based on a variation of the above problem (Problem 4.1.1). SIDH was one of the most popular post quantum alternatives due it's small key size and fast implementation and became a leading contender in the 2016 NIST Post-Quantum Cryptography Standardisation

[4]It must be noted that as of present date, we still do not have a powerful enough quantum computer which executes Shor's algorithm to break current levels of security given by RSA, ECDH etc. The rough estimate until we have such a computer is 15 years from now.

process, and made it to the 4th round. A sudden halt in it's indisputable 11 year reign came in July, 2022 when a polynomial-time classical attack was found [CD22], which was further enhanced by [MM22], [Rob22]. It must be noted that these attacks leveraged the extra information revealed by SIDH (namely image of torsion points), however these attacks do not affect other isogeny-based schemes such as CSIDH, SQISign and its variants which are built on different premises.

While this series of events led to the demise of a promising post-quantum candidate, it also led cryptographers to probe deeper into the understanding of the fundamental assumptions involved and the exploration of mathematical techniques from the theory of abelian varieties (which featured heavily in the 2022 attacks). This has also spurred a flurry of activity towards building more robust cryptographic schemes based on isogenies.

One of the popular lines of research involves coming up with countermeasures which intend to mask the crucial piece of torsional information, that enabled the SIDH attacks, and thereby aiming to have a similar secure isogeny based protocol. A few to name in this direction would be [FMP23, BMP23, BM24, NO23]. However, cryptographers by training, remain highly skeptical of new assumptions and constructions, and the security claims they make due to the possible presence of crucial flaws in such designs. The practices adopted to ensure if such flaws are indeed absent mostly fall in two categories: provable security and cryptanalysis. While the former method tries to model plausible attacks against the protocol and reduce the central assumptions involved in a cryptosystem to a known *hard* problem, the latter method often involves coming up with actual attacks against the cryptosystem in the presence (and even absence) of certain assumptions.

In this thesis, we take the latter approach of cryptanalysis of three isogeny-based key exchange protocols and SIDH countermeasures, namely: M-SIDH [FMP23], FESTA [BMP23] and POKÉ [BM24]. We attempt to construct attacks against these three schemes, in the context of 'leakage model', in which the core assumption is that partial information about the secret components is available to the adversary. In real-world deployment, side-channel attacks are a class of attacks where the adversary can learn some partial information about the internal secret states of a protocol through exploitation of the physical attributes of the computing device, (such as through its power consumption, electromagnetic radiation, timing, temperature etc.) or flaws in the system (such as imperfect deletion). The leakage model (as described in Section 5.1) formally captures these classes of attacks, and enables us to predict and attack the robustness of a cryptosystem, even before an actual side-channel is discovered.

Contributions

The main contributions to current literature on cryptanalysis of isogeny-based schemes are presented in Chapter 5, which include novel attacks against M-SIDH, FESTA, POKE involving both algebraic and combinatorial techniques. The contents of this chapter is a result of a joint research collaboration between Riccardo Invernizzi, KU Leuven, Prof. Péter Kutas, Jonas Meers, Ruhr University Bochum and the author. The results presented here are hitherto unpublished, and shall appear in full detail in a research publication in the near future.

Outline

The thesis is divided into the following thematic chapters: Chapter 2 summarizes the necessary mathematical preliminaries required to understand the thesis. Chapter 3 delves into the known algebraic cryptanalytic techniques in literature, notably Coppersmith's method and its variations. Chapter 4 summarizes the three target cryptosystems, their protocol design, underlying problems and known security analyses. Chapter 5 presents the attacks against the target cryptosystems in the context of leakage model. Chapter 6 presents a minor result obtained by the author in relation to the parameter choice of the FESTA cryptosystem.

Chapter 2

Preliminaries

In this chapter we shall be presenting the mathematical preliminaries involved to fully understand the ideas presented in this thesis. Section 2.1 is primarily sourced from [De 17, De 20, Sil92] and Section 2.2 is primarily sourced from [Gal12].

2.1 Elliptic Curves and Isogenies

Throughout this section, let k be a field and $\bar{k}$ its algebraic closure. We begin by having some preliminary definitions:

2.1.1 Elliptic curves

Definition 2.1.1. (Projective space)

We denote the projective space of dimension n by $\mathbb{P}^n$ or $\mathbb{P}^n(\bar{k})$, which is the set of all $(n+1)$-tuples

$$(x_0, \ldots, x_n) \in \bar{k}^{n+1}$$

such that $(x_0, \ldots, x_n) \neq (0, \ldots, 0)$ taken modulo the equivalence relation $(x_0, \ldots, x_n) \sim (y_0, \ldots, y_n)$ iff $\exists \lambda \in \bar{k}$ such that for all i, we have $x_i = \lambda_i y_i$.

Denote $(x_0 : \cdots : x_n)$ as the equivalence class of a given projective point.

Definition 2.1.2. (Rational points)

The set of k-rational points is defined in the following manner:

$$\mathbb{P}^n(k) = \{(x_0 : \cdots : x_n) \in \mathbb{P}^n | x_i \in k \text{ for all i}\}$$

Fixing arbitrarily $x_n = 0$, we define a projective space of dimension $n-1$, whose points we call points at infinity.

Assuming that $\text{char}(k) \neq 2, 3$, we define an elliptic curve:

Definition 2.1.3. (Weierstrass Equation)

An elliptic curve defined over k is the locus in $\mathbb{P}^2(\bar{k})$ of an equation

$$Y^2Z = X^3 + aXZ^2 + bZ^3 \tag{2.1}$$

with $a, b \in k$ and $4a^3 + 27b^2 \neq 0$. The point $(0 : 1 : 0)$ is the only point on the line $Z = 0$, and hence called the point at infinity of the curve.

If we define $x = X/Z$ and $y = Y/Z$, then the resulting expression of the Weierstrass form is called the affine form,

$$y^2 = x^3 + ax + b, \quad \text{and } O_\infty = (0 : 1 : 0)$$

One can show that for $\text{char}(k) \neq 2, 3$, any genus 1 smooth projective curve with a distinguished O_∞ is isomorphic (as an algebraic curve) to an elliptic curve defined by the Weierstrass equation, by a morphism such that $O_\infty \mapsto (0 : 1 : 0)$. We now define the group law on elliptic curves:

Definition 2.1.4. (Group law)

Let $E : y^2 = x^3 + ax + b$ be an elliptic curve. For two points $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ om E different from O_∞, we define the group operation $\oplus$ on E as follows:

1. $P \oplus O_\infty = O_\infty \oplus P = P$ for any point $P \in E$
2. If $x_1 = x_2$ and $y_1 = -y_2$, then $P_1 \oplus P_2 = O_\infty$.
3. Otherwise set

$$\text{If } P \neq Q, \quad \lambda = \frac{y_2 - y_1}{x_2 - x_1}$$

$$\text{If } P = Q, \quad \lambda = \frac{3x_1^2 + a}{2y_1}$$

then the point $(P_1 \oplus P_2) = (x_3, y_3)$

$$x_3 = \lambda^2 - x_1 - x_2$$
$$y_3 = -\lambda x_3 - y_1 + \lambda x_1$$

In the following sections we shall simply write $+$ in place for $\oplus$ and it can be shown that the above defines a commutative group under this group law. The subgroup of k-rational points on an elliptic curve E/k (i.e, defined over k) is usually denoted as $E(k)$.

Definition 2.1.5. *(m-torsion group)* For an elliptic curve E, we define the m-torsion group as:

$$E[m] = \{P \in E(\bar{k}) \,|\, mP = O_\infty\}$$

Now we shall describe the characterization of the group structure of elliptic curves. For the torsion part, it is as follows:

Theorem 2.1.6. *Let E/k and let $m \neq 0$ be an integer. The m-torsion group of E, and denoted by $E[m]$, has the following structure:*

1. *$E[m] \cong (\mathbb{Z}/m\mathbb{Z})^2$ if the characteristic of k does not divide m.*
2. *If $p > 0$ is the characteristic of k, then one of the following is true:*
   (a) *For any $i \geq 0$, $\quad E[p^i] \cong \mathbb{Z}/p^i\mathbb{Z}$*
   (b) *For any $i \geq 0$, $\quad E[p^i] \cong \{O_\infty\}$*

Proof. See [Sil92, III, Cor.6.4 and V.3, Them 3.1] □

Definition 2.1.7. [Sil92, V.3, Def.] If $\mathrm{char}(k) = p$, we define a curve E/k supersingular if $E[p^i] \cong \{O_\infty\}$ for all $i = 1, 2, 3 \ldots$ and ordinary otherwise.

Proposition 2.1.8. *(j-invariant) Let $E : y^2 = x^3 + ax + b$ be an elliptic curve, and define the j-invariant of E as*

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

Two curves are isomorphic over the algebraic closure $\bar{k}$ iff they have the same j-invariant.

Proof. See [Sil92, III, Prop. 1.4]. □

Definition 2.1.9. Let E and E' be elliptic curves defined over k. An isogeny $\phi : E \to E'$ is a non-constant algebraic map of projective varieties sending the point at infinity of E onto the point of infinity of E'.

Theorem 2.1.10. *Let E, E' be two elliptic curves, and let $\phi : E \to E'$ be an isogeny between them. Then the following conditions hold:*

1. *ϕ is a finite map of curves.*
2. *ϕ is a group morphism $(E, O_\infty, \oplus) \xrightarrow{\phi} (E', O'_\infty, \oplus)$.*
3. *ϕ has finite kernel.*

Proof. See [Sil92, III, Thm. 4.8 and Cor. 4.9]. □

We term two curves as isogenous if they have an isogeny between them. By [Sil92, III, Thm. 6.1], we can conclude that the property of being isogenous is an equivalence relation between elliptic curves. Isogenies from a curve to itself are called endomorphims. An important endomorphism is the multiplication map defined by:

$$[m] : P \mapsto mP\,; \quad \ker[m] = E[m]$$

We now shall describe the algebraic properties of isogenies in the following definitions and theorems:

Definition 2.1.11. (Degree of an isogeny)

Let $\phi : E \to E'$ be an isogeny defined over a field k, and let $k(E), k(E')$ be the function fields of E, E'. By composing ϕ with functions of $k(E')$, we obtain a subfield of $k(E)$ and we denote it by $\phi^*(k(E'))$. We define the degree of ϕ as $\deg \phi = [k(E) : \phi^*(k(E'))]$, and it is always finite.

Definition 2.1.12. An isogeny $\phi : E \to E'$ is said to be separable if the extension of the function fields is separable.

Proposition 2.1.13. *Let $\phi : E \to E'$ be a separable isogeny, then we have* $\deg \phi = |\ker \phi|$.

Proof. See [Sil92, III, Cor 4.9 and Thm.4.10]. □

All isogenies hereafter shall be considered separable. By the following proposition, we can establish that separable isogenies are determined by their kernel completely:

Proposition 2.1.14. *Let E be an elliptic curve and let G be a finite subgroup of E. There is a unique elliptic curve E', and a separable isogeny ϕ, such that* $\ker \phi = G$ *and $\phi : E \to E'$.*

Proof. See [Sil92, III, Prop. 4.12] for the proof. □

Let us denote the image curve E' as E/G.

Theorem 2.1.15. *(Dual Isogeny) Let $\phi : E \to E'$ be an isogeny of degree m. There is a unique isogeny $\hat{\phi} : E' \to E$ such that*

$$\hat{\phi} \circ \phi = [m]_E, \quad \phi \circ \hat{\phi} = [m]_{E'}$$

is called the dual isogeny and it has the following properties:

1. *$\hat{\phi}$ is defined over k iff ϕ is*

2. $\widehat{\psi \circ \phi} = \hat{\phi} \circ \hat{\psi}$ *for any isogeny* $\psi : E' \to E''$

3. $\widehat{\psi + \phi} = \hat{\psi} + \hat{\phi}$ *for any isogeny* $\psi : E \to E'$

4. $\deg \phi = \deg \hat{\phi}$

5. $\hat{\hat{\phi}} = \phi$

Proof. See [Sil92, III.6, Thm.6.2] for the proof. □.

2.1.2 Elliptic curves over finite fields

In the following subsections, we let E be an elliptic curve defined over $k = \mathbb{F}_q$. We define the following special endomorphism for curves over finite fields:

Definition 2.1.16. (Frobenius endomorphism)

For $E/\mathbb{F}_q$, we define its Frobenius endomorphism $\pi : E \to E$ such that

$$(X : Y : Z) \mapsto (X^q : Y^q : Z^q)$$

Theorem 2.1.17. *(Hasse's theorem) Let E be an elliptic curve defined over a finite field K with q elements then,*

$$||E(k)| - q - 1| \leq 2\sqrt{q}$$

Proof. See [Sil92, V, Thm.1.1]. □

We note an important theorem which connects isogenies between two elliptic curves with their Frobenius endomorphisms.

Theorem 2.1.18. *(Tate's theorem) Two elliptic curves E, E', defined over a finite field k are isogenous over k iff $|E(k')| = |E'(k')|$ for all finite extensions k' of k.*

Proof. See [Tat66, §3., Thm. 1(c)] for the full proof. □

A thorough treatment on the mathematics behind isogeny-based cryptography can be found in [De 17].

2.1.3 Pairings and Isogeny Representations

We define pairings on elliptic curves as follows, with a more detailed exposition available in [De 20].

Definition 2.1.19. (Pairings)

A pairing of two groups G_1, G_2 is a bilinear map $e : G_1 \times G_2 \to G_3$ such that:

1. $e(g^a, h) = e(g, h^a) = e(g, h)^a$
2. $e(gg', h) = e(g, h)e(g', h)$
3. $e(g, hh') = e(g, h)e(g, h')$

for all $a \in \mathbb{Z}$, all $g, g' \in G_1$, and all $h, h' \in G_2$. A pairing is said to be non-degenerate if:

- $e(g, h) = 1$, for all $g \implies h = 1$
- $e(g, h) = 1$, for all $h \implies g = 1$

A pair is said to be alternating if $G_1 = G_2$ and $e(g, g) = 1$ for all g, which implies that $e(g, h) = e(h, g)^{-1}$.

We do not define the Weil Pairing here, and instead refer to [Sil92, III.8, Def.] for its definition. However we note that the Weil pairing is a pairing as per Definition 2.1.19. An important property of the Weil pairing is as follows:

Proposition 2.1.20. *[Sil92, III.8, Prop 8.2] Let $\phi : E \to E'$ be an isogeny of elliptic curves. Then for all N-torsion points $P \in E[N]$ and $Q \in E'[N]$, we have*

$$e_N(P, \hat{\phi}(Q)) = e_N(\phi(P), Q)$$

A straightforward corollary of the above is the following:

Corollary 2.1.21. *With the assumptions of Proposition 2.1.20, then for any N, P, Q we can say that*

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

Proof. Using Proposition 2.1.20, we have that

$$e_N(\phi(P), \phi(Q)) = e_N(P, \hat{\phi} \circ \phi(Q))$$

By using Theorem 2.1.15 we have

$$e_N(P, \hat{\phi} \circ \phi(Q)) = e_N(P, [\deg \phi]Q) = e_N(P, Q)^{\deg \phi}$$

□

Definition 2.1.22. An integer is defined to be *n-smooth* if it has no prime factor greater than n.

In contemporary cryptographic literature, *smooth* integers often refer to k-smooth integers where $k \approx 10^4$.

An important lemma regarding recovering the degree if the torsion basis points and images are given, is as follows:

Lemma 2.1.23. *[FMP23, Lem.1.] Let E and E' be elliptic curves defined over $\mathbb{F}_{p^2}$ such that $\phi : E \to E'$ is an isogeny of unknown degree d and let B be a smooth integer coprime to d such that $E[B] \subset E(\mathbb{F}_{p^2})$. Set $E[B] = \langle P, Q \rangle$. Then given $P, Q, \phi(P), \phi(Q)$, there exists a polynomial time algorithm to recover d mod B.*

Proof. One computes the Weil pairing values $e_B(P, Q)$ and $e_B(\phi(P), \phi(Q)) = e_B(P, Q)^{\deg \phi}$, then one solves a discrete logarithm instance between both quantities to recover d mod B. Since $E[B] \subset E(\mathbb{F}_{p^2})$, the pairing computations run in polynomial time. Since B is smooth, then using the Pohlig-Hellman algorithm the discrete logarithm computation runs in polynomial time as well. □

We define isogeny representation as follows:

Definition 2.1.24. [BDD+24, Def.1]

Let $\phi : E_0 \to E_1$ be an isogeny defined over a finite field $\mathbb{F}_q$. An efficient representation of ϕ is some data $D \in \{0, 1\}^*$ of polynomial size in $\log(\deg \phi)$ and $\log(q)$ such that there exist:

1. an algorithm that, on input D, produces the domain and codomain E_0, E_1
2. an algorithm that, on input D, produces the degree $\deg \phi$
3. an algorithm that, on input D and a point $P \in E_0(\mathbb{F}_{q^k})$, returns $\phi(P)$ in polynomial time in $k \log(q)$ and $\log(\deg \phi)$

While in literature there are several efficient isogeny representations used in various contexts, for a complete survey one can see [Rob24]. We shall be using the following two representations in this thesis, notably in Section 4.4. We reiterate the definitions as mentioned in [BM24]. The definitions of SIDH, and FESTA are dealt with in Chapter 4.

SIDH isogenies: These isogenies are prime-power degree isogenies, with kernel generators defined over $\mathbb{F}_{p^2}$. Thus, they can be represented with a kernel representation, which consists of a curve, its domain and a single point that generates its kernel. To evaluate an SIDH isogeny $\phi : E_0 \to E_1$ on a point P, Vélu's formulas [V9char"03017] give an efficient way to obtain the codomain E_1 and its evaluation $\phi(P)$. If the isogeny

degree is l^e, a kernel generator can be expressed as a linear combination of two linearly independent points of order l^e, thus computing the pushforward of an SIDH isogeny under another secret isogeny requires revealing the action of the secret isogeny on two linearly independent points of order l^e, possibly both scaled by the same random scalar in $\mathbb{Z}_{l^e}^{\times}$.

FESTA isogenies: These isogenies have their degree written in the form $q(2^a - q)$ for some positive value a and any $q < 2^a$, and their kernel does mot have generators defined over $\mathbb{F}_{p^2}$ but over a large extension field. Hence, one needs to use a higher-dimensional representation, which includes their domain,codomain, together with their degree and action on a large torsion basis. Since the degree is of the product form $q(2^a - q)$ we can use the 2-dimensional representation. Such a representation relies on the following theorem:

Theorem 2.1.25. *(Kani's lemma) Consider the following commutative diagram of isogenies:*

$$\begin{array}{ccc} E_0 & \xrightarrow{\phi_1} & E_A \\ {\scriptstyle \phi_2}\downarrow & & \downarrow{\scriptstyle \phi_2'} \\ E_B & \xrightarrow[\phi_1']{} & E_{AB} \end{array}$$

where $\deg(\phi_i) = \deg(\phi') = d_i$ *and* $\gcd(d_1, d_2) = 1$. *Then the isogeny*

$$\Phi = \begin{pmatrix} \phi_1 & -\hat{\phi}_2' \\ \phi_2 & \hat{\phi}_1' \end{pmatrix} : E_0 \times E_0 \times E_{AB} \to E_A \times E_B$$

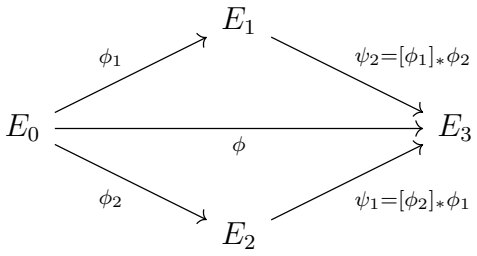
is a $(d_1 + d_2)$*-isogeny whose kernel is given by*

$$\ker \Phi = \{([-d_1]P, \phi_2' \circ \phi_1(P)) \mid P \in E_0[d_1 + d_2]\}$$

Proof. See [Kan97] for the full proof. □

Isogeny decomposition and commutative diagrams

Any isogeny $\phi : E \to E'$ of degree $d = \prod_{i=1}^{n} d_i$ can be decomposed into isogenies $\phi = \phi_n \circ \cdots \circ \phi_1$ where each ϕ_i has degree d_i. Assuming all d_i's are coprime to each other, reordering them shall lead to a different set of isogenies between the two curves. This is illustrated by the following diagram:



The above diagrams are known as isogeny commutative diagrams. For isogenies a d_1d_2-isogeny ϕ, we can decompose it as $\phi = \psi_2 \circ \phi_1 = \psi_1 \circ \phi_2$ and it can be confirmed that $\ker \psi_1 = \phi_2(\ker \phi_2)$ and converse for ψ_2. If the same diagram is defined from ϕ_1 and ϕ_2, we call ψ_1 the push-forward of ϕ_1 through ϕ_2 and denote it by $\psi_1 = \phi_{2*}\phi_1$. One can also see ϕ_1 as the pullback of ψ_1 by ψ_2 and we write it as $\phi_1 = \phi_2^*\psi_1$.

2.2 Lattice theory

In this section we shall discuss the theoretical and algorithmic aspects of lattices. A full discussion on this topic can be availed in [Gal12].

2.2.1 Basic notions

Definition 2.2.1. Let $B := \{b_1, \ldots, b_n\}$ be a linearly independent set of row vectors $\mathbb{R}^m (m \geq n)$. The lattice generated by this set is defined as

$$L := \left\{ \sum_{i=1}^{n} l_i b_i \ : \ l_i \in \mathbb{Z} \right\}$$

The set B is called a lattice basis. The lattice rank is n and the lattice dimension is m. If $n = m$ then the lattice is called a full rank lattice.

A basis matrix B of a lattice L is a $n \times m$ matrix formed by taking the rows to be basis vectors b_i. Thus $B_{i,j}$ is the j-th entry of the row b_i.

Whenever $m > n$, it is convenient to project the lattice L into $\mathbb{R}^n$ using the following construction, which captures the fact that a linear map which preserves lengths also preserves volumes.

Lemma 2.2.2. *Let B be a $n \times m$ basis matrix for a lattice L where $m > n$. Then there is a linear map $P : \mathbb{R}^m \to \mathbb{R}^n$ such that $P(L)$ is a rank n lattice and $||P(v)|| = ||v||$ for all $v \in La$. Furthermore, $\langle b_i, b_j \rangle = \langle P(b_i), P(b_j) \rangle$ for all $1 \leq i < j \leq n$.*

If the linear map is represented by a $m \times n$ matrix P so that $P(v) = vP$, then a basis matrix for the image of L under the projection P is the $n \times n$ matrix BP, which is invertible.

Proof. See [Gal12, IV, Lem.16.1.5.] for the full proof. □

The determinant of a lattice L is the volume of the parallelpiped of any basis B of a lattice L.

Definition 2.2.3. Assuming the above notations, the determinant (or volume) of a lattice L with basis matrix B is $|\det(BP)|$, where P is a matrix representing the projection of Lemma 2.2.2. When L is a full rank lattice, evidently $\det L = |\det(B)|$.

Alternatively, we have a characterization of the determinant in terms of Gram-Schmidt orthogonalisation.

Lemma 2.2.4. *Let $B := \{b_1, \ldots, b_n\}$ be an ordered basis for a lattice L in $\mathbb{R}^m$ and let $b_1^*, \ldots, b_n^*$ be the Gram-schmidt orthogonalisation of the same. Then* $\det(L) = \prod_{i=1}^{n} ||b_i^*||$.

Proof. See [Gal12, IV, Lem.16.1.14.] for the full proof. □

Definition 2.2.5. Let $B := \{b_1, \ldots, b_n\}$ be an ordered basis for a lattice L . The orthogonality defect of the basis is defined as

$$\left(\prod_{i=1}^{m} ||b_i||\right) / \det(L)$$

Definition 2.2.6. Let $L \subset \mathbb{R}^m$ be a lattice of rank n. The successive minima of L are $\lambda_1, \ldots, \lambda_n \in \mathbb{R}$ such that for $1 \leq i \leq n$, λ_i is the minimal such that there exist i linearly independent vectors $v_1, \ldots, v_i \in L$ with $||v_j|| \leq \lambda_i$ for $1 \leq j \leq i$.

We note the following important theorem by Minkowski which relates the determinant of a lattice with its successive minima.

Theorem 2.2.7. *(Minkowski) Let L be a lattice of rank n in $\mathbb{R}^n$ with successive minima $\lambda_1, \ldots, \lambda_n$ for the Euclidean norm. Then*

$$\left(\prod_{i=1}^{m} \lambda_i\right)^{1/n} < \sqrt{n} \det(L)^{1/n}$$

2.2.2 Lattice reduction

The idea of lattice basis reduction is to transform a given lattice basis into a another lattice basis which consists of vectors, short and smaller orthogonal defect. The Lenstra-Lenstra-Lovász (LLL) algorithm is an iterative algorithm which given a lattice basis performs this reduction. To achieve this we need to define a convenient notion of reduced basis, and how to proceed with the reduction.

Denote $\mu_{i,j} = \langle b_i, b_j^* \rangle / \langle b_j^*, b_j^* \rangle$, where $\{b_i^*\}$ denotes the Gram-schmidt orthogonalization of $\{b_i\}$. We define a LLL-reduced basis as follows:

Definition 2.2.8. Let $\{b_1, \ldots, b_n\}$ be an ordered basis for a lattice. Denote by $\{b_1^*, \ldots, b_n^*\}$ the Gram-schmidt orthogonalisation and write $B_i = ||b_i^*||^2 = \langle b_i^*, b_i^* \rangle$. For $1 \leq j < i \leq n$ let $\mu_{i,j}$ be defined as above. Fix $1/4 < \delta < 1$. The ordered basis is LLL-reduced (with a factor δ) if the following conditions hold:

1. $|\mu_{i,j}| \leq 1/2$ for $1 \leq j < i \leq n$.
2. (Lovász condition) $B_i \geq (\delta - \mu_{i,i-1}^2) B_{i-1}$ for $2 \leq i \leq n$.

The Lovász condition is usually set to $\delta = 3/4$

Here, we implicitly assume the Gram-Schmidt orthogonalization procedure, which performs $\mathcal{O}(n^4 m \log X^2)$ bit operations for a $n \times m$ lattice basis $B := \{b_i\}$ and $||b_i^2|| < X$. Refer to [Gal12, IV, Sec.17.3] for further details. We define the LLL algorithm in Algorithm 2.

Algorithm 2 LLL algorithm (with Euclidean norm and $\delta = 3/4$)

Input: $b_1, \ldots, b_n \in \mathbb{Z}^m$

Output: LLL-reduced basis $B_1, \ldots, b_n$

Compute the Gram-Schmidt basis $b_1^*, \ldots, b_n^*$ and coefficients $\mu_{i,j}$ for $1 \leq j < i \leq n$
Compute $B_i = \langle b_i^*, b_i^* \rangle = ||b_i||^2$ for $1 \leq i \leq n$
$k = 2$
while $k \leq n$ **do**
    for j=(k-1) downto 1 **do** ▷ Performs size reduction
        Let $q_j = \lfloor \mu_{k,j} \rceil$ and set $b_k = b_k - q_j b_j$
        Update the values $\mu_{k,j}$ for $1 \leq j < k$
    end for
    if $B_k \geq (\delta - \mu_{k,k-1}^2) B_{k-1}$ **then**
        $k = k + 1$
    else
        Swap b_k with b_{k-1}
        Update the values $b_k^*, b_{k-1}^*, B_k, B_{k-1}, \mu_{k-1,j}$ and $\mu_{k,j}$ for $1 \leq j < k$, and $\mu_{i,k}\mu_{i,k-1}$ for $k < i \leq n$.
        $k = \max\{2, k-1\}$
    end if
end while

The complexity of the above algorithm is given by the following theorem and corollary:

Theorem 2.2.9. *[Gal12, IV, Thm.17.5.1] Let L be a lattice in Z^m with a basis $b_1, \ldots, b_n$, and let $X \in Z_{\geq 2}$ be such that $||b_i||^2 \leq X$ for $1 \leq i \leq n$. Let $1/4 \leq \delta < 1$. Then the LLL algorithm 2 with factor δ terminates and performs $\mathcal{O}(n^2 \log(X))$ iterations.*

Corollary 2.2.10. *[Gal12, IV, Cor.17.5.4] Let the assumptions of Theorem 2.2.9 hold true. Then, the LLL algorithm 2 requires $\mathcal{O}(n^3 m \log(X))$ arithmetic operations on integer size of $\mathcal{O}(n \log(X))$. Using arithmetic gives us the running time of $\mathcal{O}(n^5 m \log(X)^3)$ bit operations.*

2.3 Gröbner Bases

In commutative algebra, a Gröbner basis is a special generating set of an ideal in a polynomial ring $k[x_1 \ldots x_n]$, over a field k. The Gröbner bases is useful in deducing important properties of the ideal and the associated algebraic variety, especially the dimension and the finiteness of zeroes. It is widely used in algebraic cryptanalysis, where a cryptosystem is modeled as a set of non-linear equations, and to break it, is equivalent to solving the system of equations. In this section we recall the basics of Gröbner bases in three variables.

Let $\mathbb{Z}[x, y, z]$ be the polynomial ring x, y, z over $\mathbb{Z}$. A monomial is an elementary polynomial $x^{a_1} y^{a_2} z^{a_3}$ with $a_1, a_2, a_3 \in \mathbb{N}$ and a term is $\lambda x^{a_1} y^{a_2} z^{a_3}$, with $\lambda \in \mathbb{Z}$. Let us denote (a_1, a_2, a_3) as the monomial $x^{a_1} y^{a_2} z^{a_3}$.

Definition 2.3.1. The Newton polygon of a polynomial $p \in \mathbb{Z}[x, y, z]$ is defined as the convex hull of all monomials (viewed as points in $\mathbb{N}^3$) that appear with a non-zero coefficient in p.

Definition 2.3.2. Let $\mathcal{M}$ be a set of monomials. A monomial order on $\mathcal{M}$ is a total order $\prec$, which satisfies the following properties:

1. For every $\lambda \in \mathcal{M}$, it holds that $1 \prec \lambda$.

2. If $\lambda_1 \prec \lambda_2$, then $\lambda \cdot \lambda_1 \prec \lambda \cdot \lambda_2$ for every monomial in $\mathcal{M}$.

If a monomial ordering is chosen, the initial term of a polynomial p, denoted by $LT(p)$, refers to its greatest term.

Definition 2.3.3. (Gröbner bases)

Let I be an ideal of $\mathbb{Z}[x, y, z]$, $LT(I)$ is the set of all leading terms of the polynomials which belong to I. If the set $\{q_1, \ldots, q_l\}$ is composed by polynomials of I such that $(LT(q_1), \ldots, LT(q_l)) = LT(I)$, we define it a Gröbner basis of I.

An important question to answer is when does one know that a given basis $\{q_1, \ldots, q_l\}$ is Gröbner bases. We have the Buchberger criterion to distinguish between so, and the corresponding algorithm [Buc70] to construct such bases.

Definition 2.3.4. (S-polynomial, [Buc76, Def.1.8])

Let $f, g \in \mathbb{Z}[x, y, z]$ be non-zero polynomials and let $LT(f)$ denote the leading term of a polynomial f, with respect to some monomial order. We call the least common multiple of $LT(f)$ and $LT(g)$ as $\ell = LCM(LT(f), LT(g))$. Then we define the S-polynomial of f and g as the combination:

$$S(f, g) = \frac{\ell}{LT(f)} \cdot f - \frac{\ell}{LT(g)} \cdot g$$

Then we have the following theorem:

Theorem 2.3.5. *[Buc76, Thm. 3.3] Let $I \subset \mathbb{Z}[x, y, z]$ be an ideal, then $G = \{q_1, \ldots, q_l\}$ is a Gröbner basis of I if and only if $\forall i \neq j$, the remainder upon division of $S(q_i, q_j)$ by G is zero.*

The following algorithm is based on the above criterion, which transforms a basis of an ideal I into it's Gröbner basis:

Algorithm 3 Buchberger's algorithm

Input: A set of polynomials $\{q_1, \ldots q_l\}$ that generate I.

Output: A Gröbner basis G for I

1: Set G as F

2: For every $q_i, q_j \in G$, let l_{ij} denote the LCM of $LT(q_i)$ and $LT(q_j)$.

3: Choose two polynomials in G and let us define the S-polynomial $S_{ij} = \frac{l_{ij}}{LT(q_i)} \cdot q_i - \frac{l_{ij}}{LT(q_j)} \cdot q_j$

4: Divide S_{ij} by G with multivariate division until the remainder is zero. If remainder is non-zero, then add it to G.

5: Iterate steps 1-4 until all polynomial pairs are considered in G.

6: **return** G

In practice, one can compute a Gröbner basis using the F4 algorithm [Fau99].

Chapter 3

Finding small roots of polynomials

Given a family of polynomial equations, over integer or modulo rings, the problem of finding 'small' roots has importance in cryptanalysis of many protocols relying on number-theoretic or algebraic hard problems. In this chapter we are going to look at two such methods, namely Coppersmith's method and Automated Coppersmith approach. The contents of this chapter is primarily sourced from [Cor04, May21, May09, MN23].

3.1 Coppersmith's method

The first cryptographically relevant procedure to solve the problem of finding small roots of polynomials was proposed by Don Coppersmith in the articles [Cop96b, Cop96a]. The method uses lattice reduction techniques as described in Section 2.2. This method provides provable guarantees to find all roots smaller than a given bound, in polynomial time. The polynomial run time is due to the ingenious use of the Lenstra-Lenstra-Lovász (LLL) algorithm, as discussed in Section 2.2. The method was first applied in [Cop96b, Cop96a, Cop97] which showcased some of the powerful attacks the RSA function, and since has been a staple cryptanalytic tool against hard problems. For detailed exposition on RSA cryptanalysis using this method, [May21] is a great resource.

While the above cited articles present a clean procedure in the univariate polynomial case, in the multivariable case the procedure becomes increasingly complicated. In [Cor04], Coron gave a simpler and efficient algorithm for finding small roots in the bivariate case following the simplification made in [HG97].

3.1.1 Univariate case

Given a polynomial $f(x)$ the goal is to construct a polynomial $g(x)$ of usually larger degree such that every small modular root x_0 of f i.e, $f(x_0) = 0 \bmod N$, $|x_0| < X$ is also a root of g over $\mathbb{Z}$. For some fixed $m \in \mathbb{Z}$, $g(x)$ is constructed as integer linear combination of $h_{ij} = x^j N^i f^{m-i}(x)$. Then by design $g(x_0) = 0 \bmod N^m$ as well.

If we identify polynomials $\{h_{ij}\}$ by coefficient vectors, the integer linear combinations of these vectors form a lattice L. One observes that the small vectors in L correspond to linear combinations $g(x)$ with small coefficients. Assuming $|g(x_0)| < N^m$ for all $|x_0| \leq X$, then it implies that $g(x_0) = 0$, since the only multiple smaller than N^m is $0 \cdot N^m = 0$. This implies that $g(x)$ has the desired roots over integers. The following lemma (stated in full generality) formalizes the above discussion.

Lemma 3.1.1. *[HG97] Let $g(x_1, \ldots, x_k)$ be a univariate polynomial with n monomials. Let m, X be positive integers. Suppose:*

1. *$g(x_{01}, \ldots, x_{0k}) = 0$ mod N^m where $|x_{0i}| \leq X_i \; \forall i \in \{1, \ldots, k\}$*
2. *$||g(x_1 X_1 \ldots x_k X_k)|| < N^m/\sqrt{n}$*

Then $g(x_{01}X_1, \ldots, x_{0k}X_k) = 0$ holds over the integers.

Proof. Property 2 implies that

$$\begin{aligned} |g(x_{01} \ldots x_{0k})| = |\sum_i c_i x_{01}^{j_1} \ldots x_{0k}^{j_k}| &\leq \sum_i |c_i x_{01}^{j_1} \ldots x_{0k}^{j_k}| \\ &\leq \sum_i |c_i| X^i \leq \sqrt{n}||g(x_1 X_1 \ldots x_k X_k)|| < N^m. \end{aligned}$$

By property 1 we know that $g(x_{01} \ldots x_{0k})$ is a multiple of N^m, and therefore $g(x_{01}X_1 \ldots x_{0k}X_1) = 0$. □

Recall, by Theorem 2.2.10 for a lattice L with a basis $B = \{b_i\}_{i=1\ldots n}$, the LLL algorithm outputs a vector $v \in L$ such that $||v|| \leq 2^{\frac{n-1}{4}} \det(L)^{1/n}$. To find a solution to the polynomial f, then one can implement LLL such that we satisfy property 2 of Lemma 3.1.1 i.e, one has to find v such that

$$||v|| \leq 2^{\frac{n-1}{4}} \det(L)^{1/n} < \frac{N^m}{\sqrt{N}}$$

In the following theorem we shall see that $\det(L) = N^{\Theta(m)}$ with $m \approx \log N$. For sufficiently large N we have the enabling condition $\det L \leq N^{mn}$ which optimizes the choice of bound X and plays a crucial role in all Coppersmith style constructions.

Theorem 3.1.2. *Let N be an integer of unknown factorization. Let $f(x)$ be a univariate monic polynomial of constant degree δ. Then we can find all solutions x_0 of the equation*

$$f(x) = 0 \ mod \ N \ with \ |x_0| \leq N^{1/\delta}$$

in time polynomial of $\log N$ *and* δ.

Proof. We provide a sketch of the full proof in [May09]. Choose $m \approx \frac{\log N}{\delta}$ and define the collection

$$h_{ij} = \left\{ x^j N^i f(x)^{m-i} \,|\, \forall i, 0 \leq i < m, 0 \leq j < \delta \right\}$$

Evidently, coefficient vectors of $h_{ij}(xX)$ form a $n = m\delta \approx \log N$-dimensional lattice basis such that $\det(L) \approx N^{\delta m^2/2} X^{n^2/2}$. The enabling condition described above then becomes $N^{\delta m^2/2} X^{n^2/2} \leq N^{mn}$. Using $n = m\delta$, we have $X^{\delta^2 m^2} \leq N^{\delta m^2}$, which gives us the desired root bound $X \leq N^{1/\delta}$.

The $\log N$-dimensional lattice has largest entries of bit-size $\log B_{max} = \mathcal{O}(m \log N) = \mathcal{O}(\log^2 N)$ where B_{max} is the largest basis entry.

Together with the runtime of the LLL algorithm, we have that the algorithm runs in time polynomial of $\log N$ and δ. □.

3.1.2 Bivariate case

Given an irreducible polynomial $P(x, y) = \sum_{i,j} p_{ij} x^i y^j$ with coefficients in $\mathbb{Z}$ and knowing that it has an integer root, with bounds X, Y, then the goal is to recover such a root.

Let k be a parameter whose larger size ensures the success of the algorithm. Let $a := P(0, 0)$ and $W = ||P(xX, yY)||_\infty$ where $||P(x, y)||_\infty = \max_{i,j}\{|p_{ij}|\}$. Generate $n \in \mathbb{Z}$ such that $W \leq n < 2W$ and $\gcd(a, n) = 1$. Define $q(x, y) = a^{-1} P(x, y) \bmod n$.

Consider the following set of polynomials $\mathcal{S}$: For all monomials $x^i y^j$ with $0 \leq i + j \leq k$, we have $\{q_{ij} = X^{k-i} Y^{k-j} x^i y^j q\}$. For monomials of degree $k < i, j \leq \delta + k$ we have $q_{ij}(x, y) = n x^i y^j$. By construction, for an integer root (x_0, y_0) of $P(x)$, we have $q_{ij}(x_0, y_0) = 0 \bmod n$. Define $\mathcal{M}$ as the set of all polynomials in $\mathcal{S}$ and let $|\mathcal{M}| = m$. By construction, $|\mathcal{S}| = m$. Let M_1 be a $m \times m$ matrix with columns labelled by each monomial in $\mathcal{M}$ and the rows have coefficients of polynomials in $\mathcal{S}$. Let L_1 be the lattice generated by the rows M_1. Apply LLL reduction (Theorem 2.2.10) to L_1 and let $B = \{b_1 \dots b_m\}$ be the reduced lattice basis of L_1. Construct h such that it defines the hyperplane of the lattice containing the small solutions to $P(x, y)$. Hence, h has small coefficients due to LLL reduction and $h(x_0, y_0) = 0$. Akin

to Section 3.1.1, we can use the Lemma 3.1.1 to show that if (x_0, y_0) is sufficiently small, then $h(x_0, y_0) = 0$ over the integers and easily solvable. The following lemma indicates how small the coefficients of $h(xX, yY)$ must be such that it is no longer a multiple of $P(x, y)$.

Lemma 3.1.3. *([Cor04], Lem. 3) Let $a(x, y)$ and $b(x, y)$ be two non-zero polynomials over $\mathbb{Z}$, separately of maximum degree d in x, y, such that $b(x, y)$ is a multiple of $a(x, y) \in \mathbb{Z}[x, y]$. Assume that $a(0, 0) \neq 0$ and $b(x, y)$ is divisible by a non-zero integer r such that $\gcd(r, a(0, 0)) = 1$. Then $b(x, y)$ is divisible by $r \cdot a(x, y)$ and*

$$2^{-(d+1)^2} \cdot |r| \cdot ||a||_\infty \leq ||b||$$

By the above Lemma, $h(x, y)$ and $P(x, y)$ are algebraically independent when

$$h(xX, yY)||| < 2^{-\omega} \cdot (XY)^k \cdot W$$

Since $P(x, y)$ is by assumption irreducible and $h(x, y)$ is not a multiple of $P(x, y)$, the resultant polynomial $Q(x) = \text{Resultant}_y(h, P)$ is non-trivial and $Q(x_0) = 0$. From this equation, x_0 can be recovered using any standard root-finding algorithm (such as Newton's method) and finally y_0 can be recovered by solving $P(x_0, y) = 0$. The above discussion can be summarized by the following theorems:

Theorem 3.1.4. *([Cor04], Thm. 4)*

Let $P(x, y) \in \mathbb{Z}[x, y]$ be an irreducible polynomial, of degree δ in each variable. Let X, Y be upper bounds on the integer solution (x_0, y_0), and let $W = \max_{i,j}\{|p_{ij}X^iY^j\}$. If for $\varepsilon > 0$ we have that

$$XY < W^{2/3\delta-\varepsilon}$$

then in time polynomial in $(\log W, 2^\delta)$, one can find all integer pairs (x_0, y_0) pairs such that $P(x_0, y_0) = 0\, |x_0| \leq X$ and $|y_0| \leq Y$.

Theorem 3.1.5. *([Cor04], Thm. 5) Under the hypothesis of Theorem 3.1.4, except that $P(x, y)$ has total degree δ, the bound is*

$$XY < W^{1/\delta-\varepsilon}$$

3.1.3 Multivariate case

In the case of a multivariate polynomial (for $n > 3$), there are no rigorous methods of obtaining all the small roots, and all existing methods rely on heuristics. We shall illustrate this by trying to extend the above bivariate approach to the trivariate case.

Like in Section 3.1.2 the polynomials defining the lattice L_1 are of the forms $X^{k-i}Y^{k-j}Z^{k-l}x^iy^jz^lq$ and $x^iy^jz^tn$ which evaluate (x_0, y_0, z_0) to 0 over $\mathbb{Z}_n$. One must notice that given a polynomial $P(x, y)$, the procedure in Section 3.1.2 only needs to compute a polynomial h such that it is algebraically independent from P to be able to compute the desired root for P. When given a trivariate polynomial $P(x, y, z)$, in contrast we need to compute two other trivariate polynomials h_1 and h_2 such that they are algebraically independent. The heuristic part of this procedure arises from the difficulty of guaranteeing the algebraic independence in a rigorous manner. This heuristic nature extends for more than three variables.

3.2 Automated Coppersmith

A significantly improved and optimized formulation of Coppersmith's method was proposed in [MN23] where the procedure has been almost entirely automated, thus negating the necessity of involved lattice constructions in previous formulations which need to be explicitly fine tuned using ad-hoc techniques (as detailed in the previous sections). In this method one only requires to specify which monomials are required to be included in the lattice basis, after which the procedure automatically corresponds the optimal lattice basis. An implementation of this method is provided by the authors in [MN23].

Let $Z_{\mathbb{Z}}(\mathcal{F})$ denote the set of integer for a set of polynomials $\mathcal{F} \subset \mathbb{Z}[x_1, \ldots, x_k]$, and similarly $Z_{M,X_1,\ldots X_k}$ denote the modular roots of $\mathcal{F}$ bounded by the bounds $\mathcal{F}$.

Definition 3.2.1. Let $\mathcal{M}$ be a set of monomials. A monomial order on $\mathcal{M}$ is a total order $\prec$, which satisfies the following properties:

1. For every $\lambda \in \mathcal{M}$, it holds that $1 \prec \lambda$
2. If $\lambda_1 \prec \lambda_2$, then $\lambda \cdot \lambda_1 \prec \lambda \cdot \lambda_2$ for every monomial λ in $\mathcal{M}$

For a polynomial f, the leading monomial (with respect to order $\prec$) is denoted by $\mathsf{LM}(f)$ and the coefficient of the leading monomial is denoted as $\mathsf{LC}(f)$. It is evident that

$$\mathsf{LM}(fg) = \mathsf{LM}(f)\mathsf{LM}(g)$$

$$\mathsf{LC}(fg) = \mathsf{LC}(f)\mathsf{LC}(g)$$

. We say $\mathsf{LC}(f) = 1$, then we say that f is monic.

Recall, Gröbner bases and their properties from Section 2.3. In the multivariate setting of Coppersmith, one first computes $\mathfrak{a} := (h_1 \ldots h_k) \subset \bar{\mathbb{Q}}[x_1 \ldots x_k]$ (where the polynomials are similar to h in construction in Section 3.1.1). Assuming the variety of

$\mathfrak{a}$ is zero dimensional i.e, finite solutions of the polynomial system, one can efficiently obtain $Z_{\mathbb{Z}}(h_1 \dots h_k)$. The authors define the following heuristic, since there are no provable guarantees:

Heuristic 3.2.2. The polynomials obtained from Coppersmith's method generate an ideal of a zero-dimensional variety.

To construct polynomials $h_1, \dots, h_k$, the original Coppersmith's method require a set of polynomials as input which has to satisfy difficult technical constraints and usually is constructed in an ad-hoc fashion. The following method provides an automated approach towards this construction.

3.2.1 Procedure

The core idea involved in this approach is the following definition, which allows one to replace the lattice-theoretic constructions and replace them with combinatorial constraints.

Definition 3.2.3. Let $\mathcal{M}$ be a finite set of monomials, and let $\prec$ be a monomial order on $\mathcal{M}$. A set of polynomials $\mathcal{F}$ is called $(\mathcal{M}, \prec)$-suitable if:

1. Every $f \in \mathcal{F}$ is defined over $\mathcal{M}$.
2. For every monomial $\lambda \in \mathcal{M}$ there is a unique polynomial $f \in \mathcal{F}$ with leading monomial λ (with respect to $\prec$).

If $\mathcal{F}$ is $(\mathcal{M}, \prec)$-suitable and $\lambda \in \mathcal{M}$, then we denote by $\mathcal{F}[\lambda]$ the unique polynomial $f \in \mathcal{F}$ with leading monomial λ.

The above definition now allows a reformulation of the Coppersmith's method as follows:

Theorem 3.2.4. *Suppose we are given a modulus $M \in \mathbb{N}$, polynomials $f_1, \dots, f_n \in \mathbb{Z}_M[x_1, \dots, x_k]$ and bounds $0 \leq X_1, \dots, X_k \leq M$, where $k = \mathcal{O}(1)$. Furthermore, suppose we are given an integer $m \in N$, a set of monomials $\mathcal{M}$, a monomial order $\prec$ on $\mathcal{M}$ and an $(M, \prec)$-suitable set of polynomials $\mathcal{F} \subset \mathbb{Z}_{M^m}[x_1, \dots, x_k]$ with*

$$Z_{M,X_1,\dots,X_k}(f_1, \dots, f_n) \subseteq Z_{M^m,X_1,\dots,X_k}(\mathcal{F}) \tag{3.1}$$

If the conditions

$$\prod_{\lambda \in \mathcal{M}} |\mathsf{LC}(\mathcal{F}[\lambda])| \leq \frac{M^{(m-k)|\mathcal{M}|}}{\prod_{\lambda \in \mathcal{M}} \lambda(X_1, \dots, X_k)} \tag{3.2}$$

$\log(M) \geq |\mathcal{M} \geq m$ *and* $|\mathcal{M}| \geq k$ *hold, then we can compute all* $r \in Z_{M,X_1,\dots,X_k}(f_1, \dots f_n)$ *in time polynomial in* $\deg(\mathcal{F}) \cdot \log(M)$ *under Heuristic 3.2.2 for* $k > 1$.

The proof of this theorem is provided in [MN23]. The general strategy followed is to use Lemma 3.1.1, and proceed similarly as in the above Coppersmith-style theorems. The algorithm behind Theorem 3.2.4 is as follows:

Algorithm 4 Coppersmith's method

Input: Integers $M, m \in \mathbb{N}$, polynomials $f_1, \ldots, f_n \in \mathbb{Z}_M[x_1, \ldots, x_k]$, bounds $0 \leq X_1, \ldots, X_k \leq M$, set of monomials $\mathcal{M}$, monomial order $\prec$ on $\mathcal{M}$, and a $(\mathcal{M}, \prec)$-suitable set of polynomials $\mathcal{F} \subseteq Z_{M^m}[x_1, \ldots, x_k]$, satisfying the constraints of Theorem 3.2.4.

Output: All $r \in Z_{M,X_1,\ldots,X_k}(f_1, \ldots f_n)$

1: Construct $|\mathcal{M}| \times |\mathcal{M}|$ basis matrix B, with coefficient vectors of the polynomials $\mathcal{F}[\lambda](x_1X_1, \ldots, x_kX_k)$ as columns.
2: LLL-reduce B.
3: Interpret the first k column of the resulting matrix as coefficient vectors of polynomials $h_i(X_1x_1, \ldots, X_kx_k)$.
4: Compute the Gröbner basis of $(h_1(x_1, \ldots, x_k), \ldots h_k(x_1, \ldots, x_k))$.
5: **return** all $r \in Z_{M,X_1,\ldots,X_k}(f_1, \ldots f_n)$

3.2.2 Parameter choices

We now show below the procedure of choosing an optimal $\mathcal{F}$ in an automated way, once $\mathcal{M}, m$ and $\prec$ are fixed. We also give methods to choose $\mathcal{M}, m, \prec$.

Choosing $\prec$.

The choice of $\prec$ is usually of secondary importance in Coppersmith's method, and simply choosing the lexicographic order $\prec_{\mathsf{lex}}$ will suffice in most applications.

Choosing m and $\mathcal{M}$.

Instead of choosing one fixed m and $\mathcal{M}$, we define an increasing sequence $\mathcal{M}_1 \subset \mathcal{M}_2 \subset \mathcal{M}_3 \subset \ldots$ of sets of monomials. Define

$$\mathcal{M}_i := \left\{\lambda \,|\, \lambda \text{ is a monomial of } f_1^{j_1} \ldots f_n^{j_n},\ 0 \leq j_1, \ldots, j_n \leq i\right\}$$

$$m_i := i \cdot n$$

This choice can be further optimized by the choice of the polynomials $f_1, \ldots, f_n$. The condition imposed by Equation 3.2 entails that the larger $\mathcal{M}_i$ the better Coppersmith's method performs.

Choosing $\mathcal{F}$.

After fixing $\mathcal{M}, \prec$ and m. The set $\mathcal{F}$ then has to satisfy the following three conditions:

1. It has to be the $(\mathcal{M}, \prec)$-suitable
2. It has to satisfy Equation 3.1
3. It has to satisfy Equation 3.2

To satisfy Equation 3.1, which is to simply construct $\mathcal{F}$ the shift-polynomials of the following form:

$$p_{[j_1,\ldots,j_k,i_1,\ldots,i_n]} := x_1^{j_1} \ldots x_k^{j_k} \cdot f_1^{i_1} \ldots f_n^{i_n} \ldots M^{m-(i_1+\cdots+i_n)} \tag{3.3}$$

For any appropriately chosen $j_1, \ldots, j_k, i_1, \ldots, i_n \in \mathbb{N}$, where $i_1 + \cdots + i_n \leq m$. Since for any $r \in \mathbb{Z}_{M,X_1,\ldots,X_k}(f_1, \ldots, f_n)$ we have

$$f_1^{i_1}(r) \ldots f_n^{i_n}(r) \equiv 0 \bmod M^m$$

The resulting set $\mathcal{F}$ satisfies Equation 3.1.

For satisfying Equation 3.2, notice that the right hand side of it does not depend on $\mathcal{F}$, and it simply requires the product of the leading coefficients of the polynomials in $\mathcal{F}$ is smaller than some constant. Assuming f_i's are monic, it follows that the leading coefficient of the shift polynomial is

$$\mathsf{LC}(p_{[j_1,\ldots,j_k,i_1,\ldots,i_n]}) = M^{m-(i_1+\cdots+i_n)}$$

Hence the larger the sum $i_1+\cdots+i_n$ gets, the smaller gets the leading coefficient of the corresponding shift-polynomial. Thus to satisfy Equation 3.2, the shift polynomials with as large $i_1 + \cdots + i_n$.

Finally to ensure that $\mathcal{F}$ is $(\mathcal{M}, \prec)$-suitable, include every monomial $\lambda \in \mathcal{M}$ one shift-polynomial $p_{[j_1,\ldots,j_k,i_1,\ldots,i_n]}$ in $\mathcal{F}$ such that

1. the leading monomial of $p_{[j_1,\ldots,j_k,i_1,\ldots,i_n]}$ is λ, and
2. $p_{[j_1,\ldots,j_k,i_1,\ldots,i_n]}$ is defined over $\mathcal{M}$.

From the properties of LM, it follows easily that the shift polynomials which satisfy the above conditions are of the form:

$$f_{[\lambda,i_1,\ldots,i_n]} := \frac{\lambda}{\mathsf{LM}(f_1)^{i_1} \ldots \mathsf{LM}(f_n)^{i_n}} \cdot f_1^{i_1} \ldots f_n^{i_n} \cdot M^{m-(i_1+\cdots+i_n)} \tag{3.4}$$

such that $\mathsf{LM}(f_1)^{i_1} \ldots \mathsf{LM}(f_n)^{i_n}$ divides λ, and $f_{[\lambda,i_1,\ldots,i_n]}$ is defined over $\mathcal{M}$.

Thus to construct an optimal set, we simply have to enumerate all such shift-polynomials $f_{[\lambda,i_1,\ldots,i_n]}$ and then include for every $\lambda \in \mathcal{M}$ such that the choice maximizes the sum $i_1 + \cdots + i_n$.

The above idea is formalized in the following algorithm:

Algorithm 5 Constructing an optimal set $\mathcal{F}$

Input: Set of monomials $\mathcal{M}$, monomial order $\prec$ on $\mathcal{M}$, monic polynomials $f_1, \ldots, f_n$, and an integer $m \in \mathbb{N}$.

Output: $(\mathcal{M}, \prec)$-suitable set of shift-polynomials $\mathcal{F}$, satisfying Equation 3.1, and minimizing the left hand side in Equation 3.2.

1: $\mathcal{F} := \emptyset$
2: **for** $\lambda \in \mathcal{M}$ **do**
3: Enumerate all shift-polynomials $f_{[\lambda,i_1\ldots i_n]}$ such that $\mathsf{LM}(f_1)^{i_1} \ldots \mathsf{LM}(f_n)^{i_n}$ divides λ, and $f_{[\lambda,i_1,\ldots,i_n]}$ is defined over $\mathcal{M}$.
4: Among all such $f_{[\lambda,i_1,\ldots,i_n]}$ pick one that maximizes $i_1 + \cdots + i_n$ and include it in $\mathcal{F}$
5: **end for**
6: **return** $\mathcal{F}$

Chapter 4

Isogeny-based cryptosystems

In this chapter we shall be presenting the SIDH protocol, along with the three countermeasures M-SIDH, FESTA and POKÉ. For each cryptosystem, we first describe the protocol specifications, then the security analysis and known attacks against them, and finally parameter choices for each protocol. The protocol descriptions are sourced from the original papers as cited in the respective sections.

4.1 SIDH: Supersingular Isogeny Diffie Hellman

Analogous to the Diffie Hellman Key Exchange protocol, the Supersingular Isogeny Diffie Hellman (SIDH) protocol is based on key exchange construction which was introduced by [JD11]. It was one of the leading contenders in the NIST Post-Quantum Cryptography Standardisation, and made it to the 4th round, until 2022 when it was completely broken by a series of attacks [CD22, MM22, Rob22]. While the underlying assumption turns out to be insecure, SIDH was indeed a preferred candidate due to its compact secret and public key sizes. The complete description can be found in the NIST submission of SIDH [Jao22].

4.1.1 Protocol

The SIDH protocol is a DH like key-exchange that uses torsion information to complete the following diagram:

$$\begin{array}{ccc} (E_0, P_A, Q_A, P_B, Q_B) & \longrightarrow & (E_A, \phi_A(P_B), \phi_A(Q_B)) \\ \downarrow & & \downarrow \\ (E_B, \phi_B(P_A), \phi_B(Q_A)) & \longrightarrow & E_{AB} \cong E_{BA} \end{array}$$

where A, B represents curves, torsion points for Alice and Bob respectively, and $E_{AB} \cong E_{BA}$ represents the shared secret.

The protocol specification of SIDH is as follows:

Public parameters: Let $p = 2^{e_a}3^{e_b} - 1$ be a prime for integers e_a, e_b. The starting curve is a supersingular curve $E_0/\mathbb{F}_{p^2}$, public basis points of torsion group $\langle P_A, Q_A \rangle = E[2^{e_a}]$ and $\langle P_B, Q_B \rangle = E[3^{e_b}]$.

Key generation: A supersingular isogeny key pair consists of a secret key sk_i and a public key pk_i with $i \in \{a, b\}$, denoting a for Alice and b for Bob, which is an integer sampled from $\{0, 1, \ldots, 2^{e_i - 1}\}$ (denote this space as $\mathcal{K}_i$). For efficiency reasons we represents the points P, Q using the three coordinates x_P, x_Q, x_R where $R = P - Q$. The public key for both parties is given by Algorithm 6:

Algorithm 6 $\mathsf{SIDH.isogen}_i$ computing public keys

Input: A secret key sk_i

Output: A public key pk_i

1: Set $x_S \leftarrow x_{P_i + [sk_i]Q_i}$

2: Set $(x_1, x_2, x_3) \leftarrow (x_{P_m}, x_{Q_m}, x_{R_m})$

3: **for** $j = 0$ downto $e_i - 1$ **do**

4: Compute the x portion for a 2-isogeny ▷ Replace 2 with 3 in this loop if $i = b$

$$\phi_j : E_j \to E'$$

$$(x, \_) \mapsto (f_j(x), \_)$$

such that $\ker \phi_j = \langle [2^{e_i - j - 1}]S \rangle$, where S is a point on E_i with x-coordinate x_S

5: Set $E_{i+1} \leftarrow E'$

6: Set $x_S \leftarrow f_j(x_s)$

7: Set $(x_1, x_2, x_3) \leftarrow (f_i(x_1), f_i(x_2), f_i(x_3))$

8: **end for**

9: **return** $\mathsf{pk}_i = (x_1, x_2, x_3)$

Shared key establishment: Two parties, Alice and Bob denoted by $i = \{a, b\}$ establish their shared keys in the Algorithm 7.

Encryption: Using the above techniques, we define a PKE. The encryption in the PKE is performed by the Algorithm 8, where F is a function that maps the shared secret j to bitstrings.

Decryption: The decryption method is given by the Algorithm 9.

The SIDH protocol's security is guaranteed by the following problem:

Algorithm 7 $\mathsf{SIDH.isogex}_i$ shared key establishment

Input: A secret key sk_i and a public key pk_i

Output: A shared secret $j(E'_{e_i})$

1: Compute E'_0 from pk_i

2: Set $x_S \leftarrow x_{P_i+[sk_i]Q_i}$

3: **for** $j = 0$ downto $e_i - 1$ **do**

4: Compute the x portion for a 2-isogeny ▷ Replace 2 with 3 in this loop if $i = b$

$$\phi_j : E_j \to E'$$

$$(x, \_) \mapsto (f_j(x), \_)$$

such that $\ker \phi_j = \langle [2^{e_i-j-1}]S \rangle$, where S is a point on E_i with x-coordinate x_S

5: Set $E_{i+1} \leftarrow E'$

6: Set $x_S \leftarrow f_j(x_s)$

7: **end for**

8: **return** $j(E'_{e_i})$

Algorithm 8 $\mathsf{SIDH.Enc}_i$ encryption algorithm

Input: A public key pk_b, $m \in \mathcal{M}$, $r \in \mathcal{K}_a$

Output: Ciphertexts (c_0, c_1)

1: Set $\mathsf{sk}_a \leftarrow r$

2: Set $c_0 \leftarrow \mathsf{isogen}_a(\mathsf{sk}_a)$

3: Set $j \leftarrow \mathsf{isoex}_a(\mathsf{pk}_b, \mathsf{sk}_a)$

4: Set $h \leftarrow F(j)$

5: Set $c_1 \leftarrow h \oplus m$

6: **return** (c_0, c_1)

Algorithm 9 $\mathsf{SIDH.Dec}_i$ decryption algorithm

Input: $\mathsf{sk}_b, (c_0, c_1)$

Output: m

Set $j \leftarrow \mathsf{isoex}_b(c_0, \mathsf{sk}_b)$

Set $h \leftarrow F(j)$

Set $m \leftarrow h \oplus c_1$

return m

Problem 4.1.1. *Let E_0/F_{p^2} be a supersingular curve with $p = 2^{e_a}3^{e_b} - 1$, set $E_0[2^{e_a}] = \langle P, Q\rangle$. Let $\phi : E_0 \to E'$ be an isogeny of degree 3^{e_b} and let $P' = \phi(P), Q' = \phi(Q)$. Given E_0, P, Q, E', P', Q', compute the isogeny ϕ.*

In the next section we shall show how this problem has been solved in polynomial time.

4.1.2 SIDH attack

In 2022, [CD22] published the first attack in which all proposed SIKE parameter sets in [Jao22], were broken. The same attack strategy had also been discovered parallelly by the authors of [MM22], however both these attacks could be prevented by clever choice of parameters which albeit would render the protocol in a much clumsy state. Robert utilized both these attacks in [Rob22] and developed an unconditional polynomial-time attack on all previously proposed SIDH variants. For exposition's sake we shall provide a sketch of Robert's version of these attacks, as described in [Gal22].

Robert's attack

Let E_0 be the initial curve, and we have the isogenies $\phi_A : E_0 \to E_A$ and $\phi_B : E_0 \to E_B$ such that $\deg \phi_A = 2^{e_a}$ and $\deg \phi_B = 3^{e_b}$. Assume $2^{e_a} > 3^{e_b}$.

The previous torsion-point attacks [Pet17, dQKL+21] incorporates the idea that if one can compute ϕ_B on points of order 3^{e_b} then computing discrete logarithms one can recover $\ker \phi_B$ can be recovered. Robert's attack follows the same idea.

Denote $c = 2^{e_a} - 3^{e_b}$. The following theorem, tells us that c can be written as a sum of four squares.

Theorem 4.1.2. *(Lagrange)*

Every positive integer is a sum of four integer squares.

We assume the simpler case where we take that all prime factors of the square free part of c are congruent to 1 mod 4, in which case we write $c = a_1^2 + a_2^2$, with integers a_1, a_2. Let $M = \begin{pmatrix} a_1 & a_2 \\ -a_2 & a_1 \end{pmatrix}$. Then $M^T M = (a_1^2 + a_2^2)Id_2 = c \cdot Id_2$ where Id_2 is the 2×2 identity matrix. Thus, M defines an isogeny $\alpha : E \times E \to E \times E$ for any elliptic curve E such that $\alpha(X, Y) = ([a_1]X + [a_2]Y, [-a_2]X + [a_1]Y)$. Here the dual isogeny corresponds to M^T such that $\hat{\alpha}\alpha = [c]$ as a map from E^2 to itself.

We can extend the secret isogeny $\phi_B : E_0 \to E_B$ to $E_0^2 \to E_B^2$ by defining $\phi_B(X, Y) = (\phi_B(X), \phi_B(Y))$. Then ϕ_B commutes with α, since ϕ_B is a group homomorphism. On the 4-dimensional abelian variety $\mathcal{A} = E_0^2 \times E_B^2$, one can define

an isogeny F by $F = \begin{pmatrix} \alpha & \hat{\phi_B} \\ -\phi_B & \hat{\alpha} \end{pmatrix}$. The dual isogeny is $\hat{F} = \begin{pmatrix} \hat{\alpha} & -\hat{\phi_B} \\ \phi_B & \alpha \end{pmatrix}$, such that $\hat{F}F = (c + 3^b)2^{e_A} Id_4$ where Id_4 is the identity matrix. Now, from the torsion points in SIDH, we know how ϕ_B acts on $E_0[2^{e_a}]$. Since $\hat{\phi_B}\phi_B = [3^{e_b}]$, we can also compute how $\hat{\phi_B}$ acts on $E_B[2^{e_a}]$. Hence we can compute how F acts on $\mathcal{A}[2^{e_a}]$. Since $\ker F \subseteq \mathcal{A}[2^{e_a}]$, we can compute $\ker F$. Using higher dimensional methods, we can compute the isogeny F and evaluate it on any point, and hence compute ϕ_B on $E_0[3^b]$, and break SIDH.

The attack in full generality as described in [Rob22] over an eight dimensional abelian variety $E_0^4 \times E_B^4$. The complexity of this attack is in time polynomial of $(\log q, \log n, l)$ where q is the base-field cardinality, n is the isogeny degree, and l is the largest prime factor of n. However, the complexity remains exponential to the dimension of the elliptic-curve-product, which assigns a large constant factor to the attack cost.

An alternate exposition of these attacks can be found in [Pan22].

4.2 M-SIDH: Masked Supersingular Isogeny Diffie Hellman

The Masked Supersingular Isogeny Diffie Hellman (M-SIDH) by Fouotsa et.al. [FMP23] is one of the first countermeasures against the attacks on SIDH, which crucially involve exploiting revealed images of torsion points under the secret isogeny. The crux of this protocol is to partially hide the torsion information by scaling the same with carefully chosen scalars. While the article [FMP23] introduces two protocols, namely M-SIDH and MD-SIDH, in which the latter involves hiding the degree of the secret isogeny rather than the torsional information, the authors have shown that the two protocols reduce to one another i.e, an instance of one is also an instance of another, and attacks on M-SIDH can be ported to MD-SIDH and vice-versa. In this thesis we shall be focusing on M-SIDH and it's cryptanalysis due to the presence of hidden torsion information, a technique also adapted in later cryptosystems.

4.2.1 Protocol

This protocol is a similar instantiation of the SIDH protocol with the major difference that the direct images $\phi(P), \phi(Q)$ of torsion basis points P, Q under the secret isogeny ϕ is not available to adversaries however the key exchange is still completed.

This can be achieved by scaling the images $\phi(P), \phi(Q)$ by a random integer

$\alpha \in \mathbb{Z}/B\mathbb{Z}^\times$, that is instead of revealing $\phi(P), \phi(Q)$ in the public key, one reveals $[\alpha]\phi(P), [\alpha]\phi(Q)$. However, using Lemma 2.1.23 one can recover $\alpha^2 \deg \phi$, from which one can derive $\alpha^2 \bmod B$ for a smooth prime B. Taking a square root α_0 of α, one can recover $[\alpha\alpha_0^{-1}]\phi(P), [\alpha\alpha_0^{-1}]\phi(Q)$ where $(\alpha\alpha_0^{-1})^2 = 1 \bmod B$. Hence one can sample α from the following set:

$$\mu(B) = \left\{x \in \mathbb{Z}/N\mathbb{Z} \mid x^2 = 1 \bmod N\right\} \tag{4.1}$$

Since the security against the SIDH attacks relies on hiding of torsional information, it is imperative that the adversary cannot recover the scalar α. Hence the degree of the secret isogeny is chosen such that there are an exponential number of square roots of 1 mod B. Now we describe the protocol:

Public Parameters: Let λ be the security parameter and $t = t(\lambda) \in \mathbb{N}$ be an integer depending on λ. Let $p = ABf - 1$ be a prime such that $A = \prod_{i=1}^{t} \ell_i$ and $B = \prod_{i=1}^{t} q_i$ are coprime integers ℓ_i, q_i are distinct small primes, $A \approx B \approx \sqrt{p}$ and f is a small cofactor. Let E_0 be a supersingular curve over $\mathbb{F}_{p^2}$. Let $E_0[A] = \langle P_A, Q_A\rangle$ and $E_0[B] = \langle P_B, Q_B\rangle$. The public parameters are $(E_0, p, A, B, P_A, Q_A, P_B, Q_B)$.

Key Generation (Alice): Alice uniformly samples two random integers $\alpha \xleftarrow{\$} \mu(B)$ and $a \xleftarrow{\$} \mathbb{Z}/A\mathbb{Z}$ respectively. Alice computes $E_0 \xrightarrow{\phi} E_A = E_0/\langle P_A + [a]Q_A\rangle$. Her public key is the tuple $pk_A = (E_A, R_b = [\alpha]\phi_A(P_B), S_b = [\alpha]\phi_A(Q_B))$ and her secret key $sk_A = a$.The integer α is deleted post key generation.

Key generation (Bob): Analogously, Bob uniformly samples two random integers $\beta \xleftarrow{\$} \mu(A)$ and $b \xleftarrow{\$} \mathbb{Z}/B\mathbb{Z}$ respectively. Similarly as above Bob computes $E_0 \xrightarrow{\phi} E_B = E_0/\langle P_B + [b]Q_B\rangle$. Bob's public key $pk_A = (E_A, R_a = [\beta]\phi_B(P_A), S_a = [\beta]\phi_B(Q_A))$ and his secret key is $sk_B = b$. The integer β is deleted post key generation.

Shared Key: After receiving pk_B, Alice checks if $e_A(R_a, S_a) = e_A(P_A, Q_A)^B$,and she aborts if not. Alice computes $E_B \xrightarrow{\phi'_A} E_{BA} = E_B/\langle R_a + [a]S_a\rangle$. Alice's shared key is $j(E_{BA})$. Similarly, Bob checks the pairing $e_B(R_b, S_b) = e_B(P_B, Q_B)^A$, if not he aborts. Bob computes $E_A \xrightarrow{\phi'_B} E_{AB} = E_A/\langle R_b + [b]S_b\rangle$. Bob's shared key is $j(E_{AB})$.

The underlying hard problem which guarantees security of the M-SIDH key exchange protocol is as follows:

Problem 4.2.1 (**Masked Torsion isogeny problem**). *Let $A = \prod_{i=1}^{t} \ell_i$ and $B = \prod_{i=1}^{t} q_i$ be two smooth co-prime integers, let f be a small cofactor such that $p = ABf - 1$ is prime, with $A \approx B$. For a supersingular elliptic curve $E_0/\mathbb{F}_{p^2}$, let $E_0[B] = \langle P, Q\rangle$. Let $\phi : E_0 \to E$ be a uniformly random A-isogeny and let α be a uniformly random element of $\mu(B)$. Given $E_0, P, Q, E_A, [\alpha]\phi(P), [\alpha]\phi(Q)$, compute ϕ.*

For small values of t, it is evident that Problem 4.2.1 is not hard. Since we want exponential roots of unity modulo B (and correspondingly A), it may seem fine to set $t = \lambda$ such that there shall be 2^λ square roots of unity. However in as we shall see in subsection 4.2.2, such a choice does not yield security and we need t to be larger.

4.2.2 Security Analysis

In this section we first list out the attacks to which M-SIDH is immune as listed out in [FMP23], and following that we sketch an attack presented in [CV23] which breaks some special cases M-SIDH and FESTA (Section 4.3).

Guessing enough torsion point information

In the above protocol description, Bob's isogeny has degree $B \approx A$, then one only needs the exact images of the $\sqrt{B} \approx \sqrt{A}$ torsion points to run the [CD22, MM22, Rob22] attacks. Assume we are provided with the images of the A-torsion points such that $A = {}_1 \dots {}_t$.

Let $n \geq 1$ be the largest index such that $\sqrt{B} \leq l_n \dots l_t$. Let $N = l_n \dots l_t$. Then Bob's secret isogeny ϕ_B can be recovered from its action on the N-torsion points. From the action of $[\alpha] \circ \phi_B$ on the A-torsion points, one can deduce the action of $[\alpha] \circ \phi_B$ on the N-torsion points. To apply the SIDH attacks successfully, one needs to know the secret square root of unity α. Since N has $t - n + 1$ factors, then there at most 2^{t-n+1}-factors, then there are maximum 2^{t-n+1} square roots of unity modulo N. One can hence try all these square roots of unity till one successfully applies the SIDH attacks.

The complexity of this attack is given by the following theorem:

Theorem 4.2.2. *[FMP23, Thm.7,8]*

[FMP23, Algorithm 1] is correct and runs in time $O(2^{t-n+1})$ and $O(2^{(t-n+1)/2})$ using a classical and quantum computer respectively, such that $t - n + 1 \leq t/2$.

Lollipop attack

While [CD22, MM22, Rob22] attacks seem to require the exact knowledge of torsion point images, the older torsion point attacks [BKM+21, FP22] seem to need these images upto a constant, although A is required to be much larger than B. Let $\mathcal{E} \in \mathrm{End}(E_0) : (x, y) \to (-x, iy)$ be a non-trivial automorphism of E_0 and let $\psi := \phi \circ \mathcal{E} \circ \hat{\phi}$ be the "lollipop endomorphism" constructed in Petit's attack [Pet17]. As images of torsion points under ϕ are provided upto a square root of unity α mod A, then we

have that

$$[\alpha]\phi \circ \mathcal{E} \circ \widehat{[\alpha]\phi} = [\alpha^2] \circ \phi \circ \mathcal{E} \circ \hat{\phi} = [\alpha^2] \circ \psi$$

Hence from the action of $[\alpha]\phi$ on the A-torsion, one can recover $[\alpha^2] \circ \psi$ on the A-torsion. Since $\alpha^2 = 1 \bmod A$, then the images of ψ are exact. Moreover as ψ has degree $B^2 \approx A^2$ and images of the torsion point of order A are known, one can apply [Rob22] to ψ instead of ϕ. After recovering ψ, one recover ϕ efficiently.

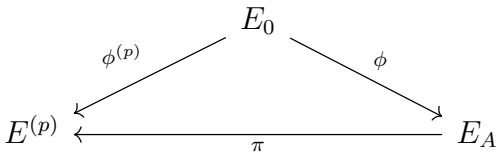
Fououtsa-Petit adaptive attack

This attack [FP22] consists of actively transforming a balanced SIDH instance ($A \approx B$) into an imbalanced one i.e, $B < A^* = NA$ where $N \approx p$, then running the torsion attacks on the imbalanced SIDH where the secret isogeny has degree B and the torsion points have order $A^* = NA$, to recover the secret isogeny. In [FP22, Section 3.2], the authors show that the torsion attacks even work while the torsion point images are scaled with an unknown scalar. To recover an isogey $\phi : E \to E'$ from its action on large enough torsion points, the attack uses the torsion point information and a suitable endomorphism θ of E to compute the endomorphism $\psi = \phi \circ \theta \circ \phi$ of E' and techniques from [Pet17] are used to recover ϕ.

[CV23] attack

The primary security argument for M-SIDH (and later FESTA) is that the polynomial time attack on SIDH no longer applies since the exact images of torsion points are not revealed, hence it is impossible to recover $\phi : E_0 \to E$ using the same attacks. The authors of [CV23] instead, recovers a related isogeny in polynomial time, in particular which does not map $E_0 \to E$. This attack is a generalisation of the lollipop attack (Section 4.2.2). Since we do not know the exact images of the torsion points due to the presence of the scalars, we construct a new isogeny ψ (related to ϕ) from E to some other curve E' that is oblivious to the unknown scalars.

Assume E_0 is $\mathbb{F}_p$ rational. Then consider the following diagram:



Here $E^{(p)}$ denotes the Frobenius conjugate of E i.e, the curve obtained by raising all the coefficients to the p-th power, and $\pi : E \to E^{(p)}$, is the Frobenius isogeny. The isogeny $\phi^{(p)}$ is the Frobenius conjugate of ϕ and satisfies $\phi^{(p)} \circ \pi_0 = \pi \circ \phi$ with π_0 being the Frobenius endomorphism on E_0. Now, consider the isogeny $\psi = \phi^{(p)} \circ \hat{\phi}$

Table 4.1: Suggested M-SIDH parameters for 128,192, 256 bits security

AES	NIST	p (in bits)	secret key	public key	compressed pk
128	level 1	5911	$\approx$ 369 bytes	4434 bytes	$\approx$ 2585 bytes
192	level 3	9832	$\approx$ 586 bytes	4103 bytes	$\approx$ 2585 bytes
256	level 5	13000	$\approx$ 812 bytes	9750 bytes	$\approx$ 5687 bytes

from $E \to E^{(p)}$ of degree d^2. Denote with $T = \lambda\phi(P)$ and $S = \lambda\phi(Q)$, the points revealed by M-SIDH, then we can show that

$$\psi \begin{pmatrix} S \\ T \end{pmatrix} = d \cdot M_{\pi_0}^{-1} \cdot \pi \begin{pmatrix} S \\ T \end{pmatrix} \tag{4.2}$$

where M_{π_0} is such that

$$\pi_0 \begin{pmatrix} S \\ T \end{pmatrix} = M_{\pi_0} \begin{pmatrix} S \\ T \end{pmatrix}$$

where it is the transpose of the matrix of π_0 acting on $E_0[N]$ with respect to the basis $\{P, Q\}$. Since all terms in Equation 4.2 is known, we can compute the image of S, T under ψ, and then apply the SIDH attack to recover ψ since in M-SIDH we have $N > d$ and thus $N^2 > \deg(\psi) = d^2$ (See Theorem [Rob22, 6.4]). If ψ is cyclic, then we can recover the kernel of $\hat{\phi}$ since in this case $\ker(\hat{\phi}) = \ker(\psi)[d]$. See [CV23] for the full discussion on this attack.

4.2.3 Parameters

Recall that the M-SIDH primes are of the form $p = ABf - 1$ where $A = l_1 \dots l_t$ and $B = q_1 \dots q_t$ are coprime integers and l_i, q_i are distinct small primes, $A \approx B \approx \sqrt{p}$ and f is a small cofactor. Let λ be the security parameter. Then for quantum security one needs $\frac{t-n+1}{2} \geq \lambda$, where n is the largest integer satisfying $\sqrt{B} \leq l_n \dots l_t$. To generate the public parameters of M-SIDH for AES-λ security (i.e, classical λ bits security and quantum $\lambda/2$ bits security). Given λ, one samples $2t$ smallest primes for $t \geq 2\lambda$, we partition them into two sets of equal size, we use the first set to get A and we use the second to get B, such that $A \approx B$. We then check the value $t - n + 1$ and if it is less than λ, we restart with a larger t. If not, we choose a cofactor f such that $p = ABf - 1$ is prime. The key sizes for AES-128,192 and 256 security levels are given in Table 4.1.

4.3 FESTA: Fast Encryption from Supersingular Torsion Attacks

The protocol FESTA: *Fast Encryption from Supersingular Torsion Attacks* by [BMP23] is a countermeasure to the SIDH attacks, which is a public key exchange protocol. The core idea is to develop a trapdoor function i.e, a function that is easy to compute in one direction, but computationally hard to invert in the other direction without access to some additional information. In this case, the SIDH attacks are employed to invert the function, which is then used to construct a PKE. While this protocol does not involve a Diffie-Hellman key exchange, like M-SIDH and SIDH, the underlying hard problem 4.3.2 which guarantees security of the PKE still relies on the hiding of the image of the torsion points under the secret isogeny.

4.3.1 FESTA trapdoor function

Definition 4.3.1. Family of trapdoor functions Let $\mathcal{X}$ and $\mathcal{Y}$ be two finite sets. A family of trapdoor functions is a triple of algorithms $(\mathsf{KeyGen}, f, f^{-1})$ such that

1. $\mathsf{KeyGen}(\lambda) \xleftarrow{\$} (\mathsf{sk}, \mathsf{pk})$: A probabilistic key generation algorithm that outputs a secret and public key for a given security parameter λ.

2. $f(\mathsf{pk}, x) \to y$: is a deterministic algorithm that, on input a public key and $x \in \mathcal{X}$, outputs $y \in \mathcal{Y}$.

3. $f^{-1}(\mathsf{sk}, y) \to x$: is a deterministic algorithm that on input a secret key and $y \in \mathcal{Y}$, outputs $x \in \mathcal{X}$.

Trapdoor functions are correct, that is $\forall(\mathsf{pk}, \mathsf{sk})$ and $\forall x \in \mathcal{X}$ we have $f^{-1}(\mathsf{sk}, f(\mathsf{pk}, x)) = x$. Such functions are also one-way, that is given a valid output y computed with $(\mathsf{pk}, \mathsf{sk})$, any probabilistic-polynomial time adversary cannot compute the input x such that $f(x, \mathsf{pk}) = y$ with probability greater than $\mathsf{negl}(\lambda)$.

Let E_0 be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$ and $\langle P_b, Q_b \rangle = E_0[2^b]$. The public key of each trapdoor function is generated by computing a secret d_A isogeny from $\phi : E_0 \to E_A$ and consist of the curve E_A, together with the image of torsion points, scaled by a matrix from the matrix set $\Gamma \in \mathcal{M}_b$ defined over $\mathbb{Z}/2^b\mathbb{Z}$. We can denote the same as $\mathcal{A}^{\mathsf{pk}} = (E_A, R_A, S_A)$ where $\begin{pmatrix} R_A \\ S_A \end{pmatrix} = \Gamma \cdot \begin{pmatrix} \phi(P_b) \\ \phi(Q_b) \end{pmatrix}$. Let $\mathsf{pk} = (E_A, R_A, S_A)$. Then we define the computation of the trapdoor function f in the following algorithm:

Algorithm 10 $f(m = (\langle K_1 \rangle, \langle K_2 \rangle, \Gamma'), \mathsf{pk})$ trapdoor function

Input: Cyclic groups $\langle K_1 \rangle \subset E_0[d_1]$ and $\langle K_2 \rangle \subset E_A[d_2]$ of order d_1 and d_2, and $\Gamma' \in \mathcal{M}_b$.

Output: $(E_1, R_1, S_1, E_2, R_2, S_2)$

1: Compute the d_1 isogeny $\phi_1 : E_0 \to E_1$ having $\langle K_1 \rangle$ as kernel.

2: Compute the d_1 isogeny $\phi_1 : E_A \to E_2$ having $\langle K_2 \rangle$ as kernel.

3: Acting with scalar multiplication compute

$$\begin{pmatrix} R_1 \\ S_1 \end{pmatrix} = \Gamma' \cdot \begin{pmatrix} \phi_1(P_b) \\ \phi_1(Q_b) \end{pmatrix} \qquad \begin{pmatrix} R_2 \\ S_2 \end{pmatrix} = \Gamma' \cdot \begin{pmatrix} \phi_2(R_A) \\ \phi_2(S_A) \end{pmatrix}$$

4: **return** $(E_1, R_1, S_1, E_2, R_2, S_2)$

Let TorAtk denote the attack procedure for breaking Problem 4.1.1 defined in Section 4.1.1 i.e, given points $P' = \phi(P)$ and $Q' = \phi(Q)$, for some unknown d-isogeny $\phi : E \to E'$, such that $\langle P, Q \rangle = E[2^b]$, $\mathsf{TorAtk}(E, P, Q, E', P', Q', d)$ outputs the isogeny ϕ. In order to invert the trapdoor function we would like to scale the torsion points R_2, S_2 on E_2 to undo the Γ-scaling. However the public points on E_2 have already been scaled by Γ', hence we need Γ, Γ' to commute, and be invertible. The authors' mention in the article that in practice the set $\mathcal{M}_b$ could be unimodular diagonal or circulant matrices[1] and the choice should not affect security. However in Chapter 6 we see that for torsion groups of odd prime power order, the usage of circulant matrices reduces to the case of usage of diagonal matrices but in the case of order 2^b we show that the choice of $\mathcal{M}_b$ as circulant matrices over $\mathbb{Z}/2^b\mathbb{Z}$ would not reduce to that of the diagonal case, which might lead to potential insecurity.

The procedure to invert the trapdoor function f is as follows:

The following problem guarantees the security of the FESTA trapdoor function, on the assumption of hardness extracting the trapdoor information from the public parameters of the function.

Problem 4.3.2. *Computational isogeny with scaled torsion (CIST)*

Let $\phi : E_0 \to E_1$ be an isogeny of smooth degree d between supersingular elliptic curves defined over $\mathbb{F}_{p^2}$, and let n be a smooth integer coprime with d.
Given the curves E_0 with a basis P_0, Q_0 of $E_0[n]$ and the curve E_1 with a basis $\Gamma(\phi(P_0), \phi(Q_0))^T$ where $\Gamma \xleftarrow{\$} \mathcal{M}_n$, compute the isogeny ϕ.

In the above problem, the output of the FESTA one-way function produces two pairs of curves and torsion points scaled by the same matrix, the correlated scaling

[1] barring the choice of trivial matrices when sampling from $\mathcal{B}$

Algorithm 11 $f^{-1}(m = (E_1, R_1, S_1, E_2, R_2, S_2)$ trapdoor inversion algorithm

Input: A tuple $(E_1, R_1, S_1, E_2, R_2, S_2)$, the trapdoor $(\Gamma \in \mathcal{M}_b, \phi_A : E_0 \to E_A)$

Output: $(\langle K_1 \rangle \langle K_2 \rangle, \Gamma')$ such that $f(m = (\langle K_1 \rangle, \langle K_2 \rangle, \Gamma'), \mathsf{pk}) = (E_1, R_1, S_1, E_2, R_2, S_2)$

1: Recover R_2', S_2' by inverting Γ and acting with scalar multiplication

$$\begin{pmatrix} R_2' \\ S_2' \end{pmatrix} = d_1 \Gamma^{-1} \cdot \begin{pmatrix} R_2 \\ S_2 \end{pmatrix}$$

2: Compute $\psi = \phi_2 \circ \phi_A \circ \hat{\phi}_1 : E_1 \to E_2$ via $\mathsf{TorAtk}(E_1, R_1, S_1, E_2, R_2', S_2', d_1 d_A d_2)$.

3: Recover $\langle K_1 \rangle$ of the d_1-isogeny $\phi_1 : E_0 \to E_1$ from ψ using ϕ_A

4: Recover $\langle K_2 \rangle$ of the d_2-isogeny $\phi_2 : E_A \to E_2$ from ψ using ϕ_A

5: Compute Γ' such that

$$\begin{pmatrix} R_1 \\ S_1 \end{pmatrix} = \Gamma' \cdot \begin{pmatrix} \phi_1(P_b) \\ \phi_1(Q_b) \end{pmatrix}$$

6: **return** $(\langle K_1 \rangle, \langle K_2 \rangle, \Gamma')$

might can potentially make invertibility easier. To guarantee the one-wayness of the FESTA function, the following problem is introduced.

Problem 4.3.3. *Computational isogeny with double scaled torsion (CIST)*[2]

Let $\phi : E_0 \to E_1$ and $\phi : E_0' \to E_1'$ be isogenies of smooth degrees d and d' respectively between supersingular elliptic curves defined over $\mathbb{F}_{p^2}$, and let n be a smooth integer coprime with d, d', and let $\Gamma \xleftarrow{\$} \mathcal{M}_n$
Given the curves E_0, E_1, E_0', E_1' with a basis P_0, Q_0 of $E_0[n]$ and basis P_0', Q_0' of $E_0'[n]$ and the points $\Gamma(\phi(P_0), \phi(Q_0))^T$ and $\Gamma(\phi'(P'), \phi'(Q'))^T$ where $\Gamma \xleftarrow{\$} \mathcal{M}_n$, compute the isogenies ϕ and ϕ'.

Having introduced the necessary computational assumptions we can prove the one-wayness of the FESTA trapdoor function.

Theorem 4.3.4. *The function $f : \mathbb{Z}/d_1\mathbb{Z} \times \mathbb{Z}/d_2\mathbb{Z} \times \mathcal{M}_b \to \mathcal{S}$, defined in Algorithm 10, is a one-way function, assuming the hardness of Problem 4.3.2 for $d = d_A$ and $n = 2^b$ and of Problem 4.3.3 for $d = d_1, d' = d_2$ and $n = 2^b$.*

Proof. In the definition of one-wayness, the attacker $\mathcal{A}$ receives the FESTA public parameters including the d_A isogenous curves E_0 and E_A and outputs E_0, E_1, P_0, P_1, Q_1 as computed by Algorithm 10 and produces ϕ_1, ϕ_2 and the matrix Γ'.
Let us replace E_A with a random starting curve. An attacker that can distinguish between the two cases, can also be a distinguisher for the decisional version of Problem 4.3.2. If any adversary can invert the FESTA trapdoor function when the curves

E_0 and E_1 are randomly generated, it can be used to solve Problem 4.3.3, since the input and outputs are the same. □

4.3.2 Protocol

Using the trapdoor function defined in the above section we have the following public-key exchange protocol.

Public Parameters: The public parameters involved here are a prime p, the curve E_0, the values d_1, d_2, d_A, b as defined above, and a description of the set $\mathcal{M}_b$, which in our case is the set of invertible diagonal matrices. We also have two random oracles, $G : \mathbb{Z}/d_2\mathbb{Z} \times \mathcal{M}_b \to \mathbb{Z}/d_1\mathbb{Z}$ and $H : \mathbb{Z}/d_1\mathbb{Z} \to \mathbb{Z}/d_2\mathbb{Z} \times \mathcal{M}_b$.

Encryption: We first evaluate G at a randomly sampled input (r, R), and use it's output (along with message m) the kernel of the isogeny ϕ_1. The isogeny ϕ_2 and the matrix Γ', which are the remaining part of the input of the trapdoor function and deterministically derived from the randomness (r, R) and the kernel ϕ_1 via the oracle H. The output of f is the ciphertext. This is formalized in Algorithm 12.

Algorithm 12 $\mathsf{FESTA.Enc}(\mathsf{pk}, m)$ encryption algorithm

Input: The public key $\mathsf{pk} = (E_A, R_A, S_A)$ and the message m to be encrypted.
Output: Ciphertext $(E_1, R_1, S_1, E_2, R_2, S_2)$

1: Sample $r \xleftarrow{\$} \mathbb{Z}/d_2\mathbb{Z}$ and $R \xleftarrow{\$} \mathcal{M}_b$.
2: Write $m' = m||0^k \bmod d_1$ and compute $s = m' + G(r, R)$.
3: Write $(x, X) = H(s)$ and compute $s = m' + G(r, R)$.
4: Compute $\mathsf{ct} = f(s, t, T, \mathsf{pk})$ ▷ Using Algorithm 10
5: **return** $(E_1, R_1, S_1, E_2, R_2, S_2)$

Decryption: The trapdoor information is used to recover the isogenies ϕ_1, ϕ_2 and the matrix Γ', from which the message can be extracted, similarly as in the trapdoor inversion algorithm. This is formalized in Algorithm 13.

4.3.3 Security Analysis

In this section we are going to list out some the possible attacks against the FESTA protocol.

First, we consider the standard SIDH attack, considering [Rob22] attack i.e, given a d-isogeny ϕ, it is possible to recover ϕ if the image of the N-torsion is available, provided $N^2 > 4d$. While FESTA does not reveal the direct images, one can recover the determinant of the scalar matrix through pairing computation $e([\alpha]\phi(P), [\beta]\phi(Q)) = e(P, Q)^{\alpha\beta \deg \phi}$. and $P, Q, \deg \phi$ are known. This information can be used to reduce one

Algorithm 13 $\mathsf{FESTA.Dec}(\mathsf{sk}, ct)$ decryption method

Input: The secret key $\mathsf{sk} = (\Gamma, \phi_A)$ and the ciphertext $(E_1, R_1, S_1, E_2, R_2, S_2)$.

Output: The decrypted message m or $\perp$ on failure.

1: Compute $(s, t, T) = f^{-1}(\mathsf{sk}, \mathsf{ct})$ ▷ Using Algorithm 11
2: Write $(x, X) = H(s)$ and compute $r = t - x, R = X^{-1}T$.
3: Compute $m' = s - G(r, R)$ and write $m||m_k = m'$ where $|m_k| = k$.
4: **if** $m_k = 0^k$ **then**
5: **return** m
6: **else**
7: **return** $\perp$
8: **end if**

variable among the unknowns: given $P' = [\alpha]\phi(P), Q' = [\beta]\phi(Q)$, and $\alpha\beta$, scaling Q' by $(\alpha\beta)^{-1}$ mod N yields the point $Q'' = [1/\alpha]\phi(Q)$. Thus P' and Q'' are the images of P, Q scaled by a unimodular determinant. However this does not affect security majorly as α is sampled from an exponentially large set (that is, its security is akin to that of M-SIDH as we previously discussed).

The other major attack against the FESTA protocol is the [CV23] attack, as outlined in Section 4.2.2. The same attack follows and breaks FESTA for curves defined over $\mathbb{F}_p$ with the minor difference that Equation 4.2 is generalized to

$$\psi \begin{pmatrix} S \\ T \end{pmatrix} = d \cdot D \cdot M_{\pi_0}^{-1} \cdot D^{-1} \cdot \pi \begin{pmatrix} S \\ T \end{pmatrix}$$

with D is the diagonal matrix with scalars α, β as entries. Since $\alpha \neq \beta$, however the above equation does not reduce to Equation 4.2, unless the expression $D \cdot M_{\pi_0}^{-1} \cdot D^{-1}$ reduces to $M_{\pi_0}^{-1}$, i.e, until M_{π_0} is a diagonal matrix, and its entries P, Q are eigenvectors. Assuring that condition, we can proceed with the attack outlined in Section 4.2.2.

4.3.4 Parameters

For FESTA, the authors recommend $m_1, d_1, m_2, d_2, d_{A,1}, d_{A,2}$, to be odd, so that the isogenies have degree co-prime with the torsion points order. The isogeny degrees $d_1, d_A = d_{A,1}d_{A,2}$ and d_2 are pairwise coprime and sufficiently long, $\log d_1, \log d_A, \log d_2 \geq 2\lambda$, to prevent meet-in-the-middle and torsion-guess attacks. The number of solutions and the protocol efficiency depend crucially on the smoothness of degrees of the isogenies involved. Let us denote our smoothness bound as B. Let c be a positive integer such that the number $T := 2^c - 1$ is B-smooth. We begin by finding primitive solu-

tions for the equation $x^2 + y^2T = 2^b$. Given a solution (x, y) for some even $b > 0$, we have

$$y^2T = (2^{b/2} - x)(2^{b/2} + x)$$

Define T_1 to be the B-smooth part of $2^{b/2} - x$ and T_2 to be the B-smooth part of $2^{b/2} + x$. In particular we have

$$m_1^2T_1 + m_2^2T_2 = 2^{b/2+1}$$

The authors recommend $T_1T_2 > 2^{6\lambda}$, we define d_i to be the smoothest factor of T_i such that $d_i \sim 2^{2\lambda}$ for $i = 1, 2$. Additionally define $d_{A,i}$ to be the smoothest part of T_i/d_i such that $d_{A,1}d_{A,2} > 2^{2\lambda}$. Thus we have found a valid set of parameters required for FESTA.

The authors provide the following parameter set for AES-128 bits security:

$$\begin{aligned}
b &:= 632,\\
d_1 &:= (33 \cdot 19 \cdot 29 \cdot 37 \cdot 83 \cdot 139 \cdot 167 \cdot 251 \cdot 419 \cdot 421 \cdot 701 \cdot 839 \cdot 1009 \cdot 1259 \cdot 3061 \cdot 3779)^2,\\
d_2 &:= 7 \cdot (52 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 41 \cdot 43 \cdot 71 \cdot 89 \cdot 127 \cdot 211 \cdot 281 \cdot 503 \cdot 631 \cdot 2309 \cdot 2521 \cdot 2647 \cdot 2729)^2,\\
d_{A,1} &:= (59 \cdot 6299 \cdot 6719 \cdot 9181)2,\\
d_{A,2} &:= (3023 \cdot 3359 \cdot 4409 \cdot 5039 \cdot 19531 \cdot 22679 \cdot 41161)^2,\\
m_1 &:= 1492184945093476592520242083925044182103921,\\
m_2 &:= 256173313364299393300166693069,\\
f &:= 107.
\end{aligned}$$

The values d_1, d_2 are 2^{12}-smooth, while $d_A = d_{A,1}d_{A,2}$ is 2^{16}-smooth. The corresponding prime, defined as $p = 2^b d_1(d_{A,1}d_{A,2})d_2 f - 1$ is 1292-bit long. The public key and ciphertext sizes are, respectively, 561 and 1,122 bytes. The same procedure can be used while producing parameter sets for higher AES security levels.

The authors recommend compressing torsion points by expressing them in terms of linear coefficients of canonical bases, as proposed in [NR19], to reduce the bandwidth of FESTA. Since unlike in SIDH, the protocol needs exact torsion images. This means the points cannot be scaled, and their representation requires four coefficients of size equal to their order.

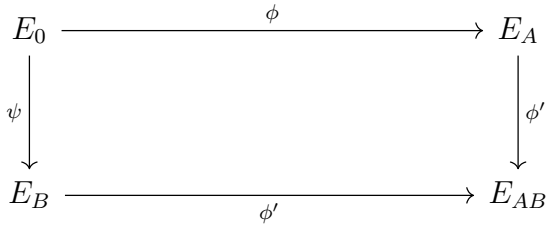
4.4 POKÉ: POint-based Key Exchange

The POKÉ protocol [BM24] is a recent isogeny based public-key exchange protocol, which also is a countermeasure to the SIDH attacks. The PKE design involves using

two different isogeny representations, the first of them being the torsion point representation where an isogeny is represented by two sets of curves and torsion points with known (smooth) degrees such that the isogeny can be evaluated by having the knowledge of curves and action of it on the torsion group. Isogenies using such representation have their kernel generators defined over $\mathbb{F}_{p^2}$. The second isogeny representation used is the two dimensional representation where both the domain and codomain of the isogeny is provided. Such isogenies are defined over large extension fields of $\mathbb{F}_{p^2}$. The core idea is that in the PKE, one party computes isogenies of unknown and non-smooth degree with kernel generators defined over large extension fields, while the other party computes smooth degree isogenies to establish a commutative diagram to obtain a shared secret.

4.4.1 Protocol

The POKÉ protocol has the following commutative diagram:



Let p be a prime of the form $p = 2^a 3^b 5^c f - 1$, where f is a small cofactor. Let E_0 be the supersingular elliptic curve defined over $\mathbb{F}_{p^2}$ with j-invariant 1728. Let $(P_0, Q_0), (R_0, S_0), (X_0, Y_0)$ denote bases of $E_0[2^a]$, $E_0[3^b]$ and $E_0[5^c]$, respectively.

Key Generation: Here we sample $q \in [1, 2^a - 1]$, coprime with 2,3,5, and compute a uniformly random isogeny $\phi : E_0 \to E_A$ of degree $q(2^a - q)$. The secret key consists of representation of ϕ, while the public key consists of E_A with the following points:

1. $P_A, Q_A := [\alpha_2]\phi(P_0), [\beta_2]\phi(Q_0)$, where $\alpha_2, \beta_2 \xleftarrow{\$} \mathbb{Z}_{2^a}^{\times}$
2. $R_A, S_A := [\gamma_3]\phi(R_0), [\gamma_3]\phi(S_0)$, where $\gamma_3 \xleftarrow{\$} \mathbb{Z}_{3^b}^{\times}$
3. $X_A, Y_A := [\delta_5]\phi(X_0), [\delta_5]\phi(Y_0)$, where $\delta_5 \xleftarrow{\$} \mathbb{Z}_{5^c}^{\times}$

One must note that the 2^a-torsion is scaled diagonally with different scalars for P_0 and Q_0, while same coefficients are used for R_0, S_0 and X_0, Y_0. The reasoning behind this is that scaling the 3^b torsion with same coefficient is needed to compute parallel isogenies during encryption, while diagonal scaling of the 2^a guarantees security of the scheme.

To generate the secret isogeny, we first generate an endomorphism θ of E_0 of degree $q(2^a - q)3^b5^c$ by using the algorithm FullRepresentInteger [DLLW23, Algorithm 1]. Then we compute the 3^b5^c-isogeny $\eta : E_0 \to E_A$ that backtracks θ, i.e, η is such that $\eta \circ \theta = [3^b5^c]\phi$, for some $q(2^a - q)$-isogeny $\phi : E_0 \to E_A$. We evaluate points under ϕ to obtain the torsion points in the public key. Since $\eta \circ \theta = [3^b5^c]\phi$, we have $\phi = [1/(3^b5^c)]\eta \circ \theta$, which allows us to evaluate ϕ on any point of order coprime with 3^b5^c. Hence we obtain images $\phi(P_0), \phi(Q_0)$ on the 2^a-torsion and thus we have a higher dimensional representation of ϕ. Using Kani's Lemma 2.1.25 we can evaluate ϕ on any points. This is summarized in the following algorithm.

Algorithm 14 Generating a $q(2^a - q)$-isogeny

Input: A degree q, a prime p of the form $p = 2^a3^b5^cf - 1$.

Output: A representation of $q(2^a - q)$-isogeny $\phi : E_0 \to E_A$.

1: Generate an endomorphism θ of E_0 of degree $q(2^a - q)3^b5^c$.
2: Compute $P', Q' = \theta(P_0), \theta(Q_0)$
3: Let $K = \ker(\hat{\theta}) \cap E_0[3^b5^c]$
4: Compute $\eta : E_0 \to E_A$ with kernel K
5: Compute $P_A, Q_A = [1/(3^b5^c)]\eta(P'), [1/(3^b5^c)]\eta(Q')$
6: Compute Φ with kernel $\langle([-q]P_0, P_A), ([-q]Q_0, Q_A)\rangle$
7: **return** Φ

Here we note that in the article [BM24] there is an additional method of key generation which is a rigorous one, however much more computationally intensive but yields uniformly distributed keys. However the method presented in this thesis is much more efficient and it's output is computationally indistinguishable from uniform distribution.

Encryption: The sender samples a random integer $r_3 \xleftarrow{\$} \mathbb{Z}_{3^b}$ and computes the parallel isogenies $\psi : E_0 \to E_B$ and $\psi' : E_A \to E_{AB}$, respectively with kernel $\omega_2 \xleftarrow{\$} \mathbb{Z}_{2^a}^\times$ as well as random unimodular matrix $\Gamma_5 \xleftarrow{\$} SL_2(\mathbb{Z}_{5^c})$. Then, the sender computes the points

$$\begin{pmatrix} P_B \\ Q_B \end{pmatrix} = \begin{pmatrix} \omega_2 & 0 \\ 0 & 1/\omega_2 \end{pmatrix} \begin{pmatrix} \psi(P_0) \\ \psi(Q_0) \end{pmatrix} \quad \begin{pmatrix} X_B \\ Y_B \end{pmatrix} = \Gamma_5 \begin{pmatrix} \psi(X_0) \\ \psi(Y_0) \end{pmatrix}$$

on the curve E_B and similarly obtains

$$\begin{pmatrix} P_{AB} \\ Q_{AB} \end{pmatrix} = \begin{pmatrix} \omega_2 & 0 \\ 0 & 1/\omega_2 \end{pmatrix} \begin{pmatrix} \psi(P_A) \\ \psi(Q_A) \end{pmatrix} \quad \begin{pmatrix} X_{AB} \\ Y_{AB} \end{pmatrix} = \Gamma_5 \begin{pmatrix} \psi(X_A) \\ \psi(Y_A) \end{pmatrix}$$

on the curve E_{AB}.

The ciphertext consists of the curve E_B, together with the points (P_B, Q_B), (X_B, Y_B), the curve E_{AB}, together with the points P_{AB}, Q_{AB}, and the message-encoding component $ct' = m \oplus KDF(X_{AB}, Y_{AB})$, where KDF is a fixed key-derivation function, often modelled as a symmetric cipher or a hash function. The encryption procedure is summarized in the following algorithm:

Algorithm 15 POKE.Enc encryption algorithm

Input: A message m, a public key $E_A, P_A, Q_A, R_A, S_A, X_A, Y_A$.

Output: A ciphertext ct.

1: Sample a random $r_3 \xleftarrow{\$} \mathbb{Z}_{3^b}$
2: Compute the isogeny $\psi : E_0 \to E_B$ with kernel $\langle R_0 + [r_3]S_0 \rangle$.
3: Compute the isogeny $\psi' : E_A \to E_{AB}$ with kernel $\langle R_A + [r_3]S_A \rangle$.
4: Sample a random $\omega_2 \xleftarrow{\$} \mathbb{Z}_{2^a}^{\times}$.
5: Compute $P_B = [\omega_2]\psi(P_0), Q_B = [1/\omega_2]\psi(Q_0)$.
6: Compute $P_{AB} = [\omega_2]\psi'(P_A), Q_{AB} = [1/\omega_2]\psi'(Q_A)$.
7: Sample $\Gamma_5 \xleftarrow{\$} SL_2(\mathbb{Z}_{5^c})$.
8: Compute $[X_B, Y_B]^T = \Gamma_5[\psi(X_0), \psi(Y_0)]^T$.
9: Compute $[X_{AB}, Y_{AB}]^T = \Gamma_5[\psi'(X_A), \psi(Y_A)]^T$.
10: Compute $ct' = KDF(X_{AB}, Y_{AB}) \oplus m$.
11: **return** ct $= ((E_B, P_B, Q_B, X_B, Y_B), (E_{AB}, P_{AB}, Q_{AB}), ct')$

Decryption: First, the receiver computes the points $P'_{AB}, Q'_{AB} = [1/\alpha_2]P_{AB}, [1/\beta_2]Q_{AB}$. By commutativity of the diagram $\phi'\psi = \psi'\phi$, it follows that P'_{AB}, Q'_{AB} are the exact images of P_B, Q_B under ϕ'. The sender computes the 2-dimensional isogeny $\Phi' : E_B \times E_{AB} \to F \times F'$ such that

$$\ker \Phi' = \langle ([-q]P_B, P'_{AB}), ([-q]Q_B, Q'_{AB}) \rangle$$

which by Kani's Lemma (2.1.25) can be expressed in the following form:

$$\Phi' = \begin{pmatrix} \phi_1 & -\hat{\phi}_2 \\ * & * \end{pmatrix} : E_B \times E_{AB} \to F \times F'$$

where $\phi_1 : E_B \to F$ and $\phi_2 : F \to E_{AB}$ are a decomposition of ϕ' i.e, $\phi' = \phi_2 \circ \phi_1$ with $\deg \phi_1 = q$ and $\deg \phi_2 = 2^a - q$. The receiver first evaluates

$$(X'_B, \_) = \Phi'(X_B, \mathcal{O}_{E_{AB}}) \text{ and } (Y'_B, \_) = \Phi'(Y_B, \mathcal{O}_{E_{AB}})$$

to obtain images of X_B, Y_B on the middle curve F, then generate a basis U, V of $E_{AB}[5^c]$ and similarly map it to the middle curve by computing

$$(U', \_) = \Phi'(\mathcal{O}_{E_B}, U) \text{ and } (V', \_) = \Phi'(\mathcal{O}_{E_B}, V)$$

The receiver then finds a change of basis matrix that maps X'_B, Y'_B to U', V', i.e, they find coefficients x, y, w, z such that

$$X'_B = [x]U' + [y]V', \quad Y'_B = [w]U' + [z]V'$$

Lastly receiver obtains $\phi'(X_B)$ as $[2^a - q]([x]U + [y]V)$ and $\phi'(Y_B)$ as $[2^a - q]([w]U + [z]V)$, from which they obtain $X_{AB} = [\delta_5]\phi'(X_B)$ and $Y_{AB} = [\delta_5]\phi'(Y_B)$. The decryption algorithm is summarized in the following algorithm:

Algorithm 16 POKE.Dec decryption algorithm

Input: $\mathsf{sk} = (q, \alpha_2, \beta_2, \delta_5), \quad \mathsf{ct} = ((E_B, P_B, Q_B, X_B, Y_B), (E_{AB}, P_{AB}, Q_{AB}, \mathsf{ct}'))$

Output: A message M'

1: Compute $P'_{AB}, Q'_{AB} = [1/\alpha_2]P_{AB}, [1/\beta_2]Q_{AB}$.
2: Compute Φ' such that $\ker \Phi' = \langle([-q]P_B, P'_{AB}), ([-q]Q_B, Q'_{AB})\rangle$.
3: Evaluate $(X'_B, \_) = \Phi'(X_B, \mathcal{O}_{E_{AB}})$.
4: Evaluate $(Y'_B, \_) = \Phi'(Y_B, \mathcal{O}_{E_{AB}})$
5: Generate a basis $\langle U, V\rangle = E_{AB}[5^c]$
6: Evaluate $U', \_) = \Phi'(\mathcal{O}_{E_B}, U)$
7: Evaluate $(V', \_) = \Phi'(\mathcal{O}_{E_B}, V)$
8: Find x, y such that $X'_B = [x]U' + [y]V'$.
9: Find w, z such that $Y'_B = [w]U' + [z]V'$.
10: Compute $X_{AB} = [\delta_5]\phi'(X_B)$
11: Compute $Y_{AB} = [\delta_5]\phi'(Y_B)$
12: Compute $m' = KDF(X_{AB}, Y_{AB}) \oplus \mathsf{ct}'$
13: **return** m'

The security of this protocol is guaranteed by the computational hardness of the following problem:

Problem 4.4.1. *C-POKE*

Let p be a prime of the form $p = 2^a 3^b f - 1$. Let E_0 be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$, and write P_0, Q_0 for a basis of $E[2^a]$, R_0, S_0 for a basis of $E[3^b]$, and X_0, Y_0 for a basis of $E_0[5^c]$. Let $\phi_A : E_0 \to E_A$ be an isogeny of degree $q(2^a - q)$ for some unknown value q. Write $P_A, Q_A = [\alpha_2]\phi_A(P_0), [\beta_2]\phi_A(Q_0)$, $R_A, S_A = [\gamma_3]\phi_A(R_0), [\gamma_3]\phi_A(S_0)$ and $X_A, Y_A = [\delta_5]\phi_A(X_0), [\delta_5]\phi_A(Y_0)$, where $\alpha_2, \beta_2, \gamma_3, \delta_5 \xleftarrow{\$} \mathbb{Z}^*_{2^a} \times \mathbb{Z}^*_{2^a} \times \mathbb{Z}^*_{3^b} \times \mathbb{Z}^*_{5^c}$.

Let $\phi_B : E_0 \to E_B$ be an isogeny of degree 3^b and write $\phi'_B : E_A \to E_{AB}$ for the pushforward $\phi_{A*}\phi_B$. Write

1. $P_B, Q_B = [\omega_2]\phi_B(P_0), [1/\omega_2]\phi_B(Q_0)$
2. $P_{AB}, Q_{AB} = [\omega_2]\phi'_B(P_A), [1/\omega_2]\phi'_B(Q_A)$
3. $X_B, Y_B = \Gamma_5[\phi_B(X_0), \phi_B(Y_0)]^T$
4. $X_{AB}, Y_{AB} = \Gamma_5[\phi'_B(X_A), \phi'_B(Y_A)]^T$

where $\omega_2 \xleftarrow{\$} \mathbb{Z}^*_{2^a}$ *and* $\Gamma \xleftarrow{\$} SL_2(\mathbb{Z}_{5^c})$.

Given $(E_0, (P_0, Q_0), (R_0, S_0))$, $(E_A, (P_A, Q_A), (R_A, S_A), (X_A, Y_A))$, $(E_B, (P_B, Q_B), (X_B, Y_B))$ *and* $(E_{AB}, (P_{AB}, Q_{AB}))$, *compute* X_{AB}, Y_{AB}

4.4.2 Security Analysis

The security of the protocol relies on the new assumption Problem 4.4.1. From it's structure, it is a Computational Diffie Hellman (CDH)-like problem where given public data produced by two parties, it computes their shared secret. The SIDH attacks [CD22, MM22, Rob22], while successful given torsional information, reasonably cannot work in this case since the Problem 4.4.1 does not reveal any action of any isogeny on any point of order 5^c on E_{AB}, thus it seems to be hard to be able to obtain the shared secret points X_{AB} without being able to evaluate either ϕ or ψ.

The authors claim that the security of the receiver against key-recovery attacks depend on the following problem:

Problem 4.4.2. *Let p be a prime of the form $p = 2^a 3^b f - 1$. Let E_0 be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$, and write P_0, Q_0 for a basis of $E[2^a]$, R_0, S_0 for a basis of $E[3^b 5^c]$. Let $\phi_A : E_0 \to E_1$ be an isogeny of degree $q(2^a - q)$ for some unknown value q. Write*

$$[P_1, Q_1]^T = \begin{bmatrix} \alpha_2 & 0 \\ 0 & \beta_2 \end{bmatrix} [\phi_A(P_0), \phi_A(Q_0)]^T \quad [R_1, S_1]^T = \begin{bmatrix} \gamma & 0 \\ 0 & \gamma \end{bmatrix} [\phi_A(R_0), \phi_A(S_0)]^T$$

where $\alpha_2, \beta_2, \gamma \xleftarrow{\$} \mathbb{Z}^*_{2^a} \times \mathbb{Z}^*_{2^a} \times \mathbb{Z}^*_{3^b} \times \mathbb{Z}^*_{5^c}$.
Given $(E_0, (P_0, Q_0), (R_0, S_0))$ *and* $(E_1, (P_1, Q_1), (R_1, S_1))$, *recover* ϕ

This problem is in fact similar to Problem 4.3.3, and hence the attacks against FESTA described in Section 4.3 can be ported easily in this case, however in this case the degree is unknown, hence it can be considered a bit stronger assumption than the Problem 4.3.3.

Analogously, the problem guaranteeing the security of the sender can be defined like above:

Table 4.2: Suggested POKE parameters for 128,192, 256 bits security

AES	NIST	p (in bits)	ciphertext	public key	compressed pk
128	level 1	431	$\approx$ 648 bytes	324 bytes	$\approx$ 270 bytes
192	level 3	648	$\approx$ 972 bytes	486 bytes	$\approx$ 405 bytes
256	level 5	863	$\approx$ 1296 bytes	648 bytes	$\approx$ 540 bytes

Problem 4.4.3. *With the same setup as Problem 4.4.1, given 4$(E_0, (P_0, Q_0), (X_0, Y_0))$, $(E_A, (P_A, Q_A), (X_A, Y_A))$, $(E_B, (P_B, Q_B), (X_B, Y_B))$ and $(E_{AB}, (P_{AB}, Q_{AB}))$, compute the isogeny ψ*

4.4.3 Parameters

To guarantee the security of the protocol, the authors make the following parameter choices:

- $a \approx \lambda$: This makes brute forcing q hard.
- $3^b \approx 2^{2\lambda}$: Choosing 3^b the degree of the encryption isogenies leads to a fast encryption procedure. The choice of b also prevents meet-in-the-middle attacks to recover ϕ.
- $5^c \approx 2^{\lambda/3}$: This enables a quick evaluation of isogenies with a higher dimensional representation [BM24, Rem. 6]. and large enough to avoid brute force attacks.

The key sizes provided by the authors are given in the Table 4.2. It is evident, that POKE is one of the most compact isogeny based protocols, due to the huge reduction in key sizes in comparison to M-SIDH and FESTA.

Chapter 5

Cryptanalysis in the bounded-leakage model

In this chapter we shall be discussing the cryptanalytic attempts made against the isogeny-based cryptosystems M-SIDH, FESTA, POKÉ. We begin by an initial exposition on leakage models in Section 5.1. In the subsequent sections, we first formalize the notion of breaking these protocols in the leakage model, then delve into the attacks in Sections 5.3 and 5.4.

5.1 Bounded-leakage model

A central line of investigation in the field of cryptanalysis is to test the robustness of assumptions which guarantee security, against feasible attacks on the cryptosystem. In real-world deployment, side-channel attacks involve attacks where the adversary can learn some partial information about the internal secret states of a protocol through exploitation of the physical attributes of the computing device, such as through its power consumption, electromagnetic radiation, timing, temperature and so on. Another source of information leakage are imperfect deletion, where the contents of memory are not properly erased and information becomes available to the adversary. Besides being of theoretical interest, study of such attacks can reveal flaws in the design of protocols and thus give rise to better guiding principles in protocol construction.

There exists several security models for leakage scenarios in literature, differing primarily on what information can become available to the adversary. One such model is the Bounded-leakage model [HLWW13], where an attacker can learn arbitrary information about the secret key, as long as the total number of bits learned is bounded by some parameter k, called the leakage bound. We formalize this notion by giving

access to the adversary an oracle O, which she can query in arbitrary order during the running of the protocol, where each query to the oracle consists of a leakage function Leak and the oracle O responds with information about the secret key, bounded by the size k. Throughout this chapter we shall focus on this model, due to it's simplicity, applicability and relevance to real world scenarios.

5.2 Problem statement

In this section we formally define the problem that we aim to solve in the leakage model. Furthermore, we reduce the initial problem of recovering secret scalars to the easier task of recovering the secret scalar modulo a sufficiently large subgroup. We then solely focus on solving this easier problem in the rest of the chapter, using the methods we have developed in the earlier chapters.

Problem 5.2.1. *Let $\phi : E \to E'$ be an isogeny of unknown degree q between supersingular elliptic curves. Further, let P, Q be a basis of $E[N]$ and $\alpha, \beta \in (\mathbb{Z}/N\mathbb{Z})^\times$ be secret scalars. Lastly, let $k \leq \log(N)$ and let $\mathsf{Leak}_k : (\mathbb{Z}/N\mathbb{Z})^\times \to \{0,1\}^k$ be a leakage function. Given the tuple*

$$(P, Q, [\alpha]\phi(P), [\beta]\phi(Q), \mathsf{Leak}_k(\alpha), \mathsf{Leak}_k(\beta)),$$

recover ϕ.

The above problem reduces to the following, seemingly easier problem, due to the existence of SIDH attacks which can be applied, once the scalars are known.

Problem 5.2.2. *Let $\phi : E \to E'$ be an isogeny of unknown degree q between supersingular elliptic curves. Further, let P, Q be a basis of $E[N]$ and $\alpha, \beta \in (\mathbb{Z}/N\mathbb{Z})^\times$ be secret scalars. Lastly, let $k \leq \log(N)$ and let $\mathsf{Leak}_k : (\mathbb{Z}/N\mathbb{Z})^\times \to \{0,1\}^k$ be a leakage function. Given the tuple*

$$(P, Q, [\alpha]\phi(P), [\beta]\phi(Q), \mathsf{Leak}_k(\alpha), \mathsf{Leak}_k(\beta)),$$

recover α, β mod M for some sufficiently large $M \leq N$.

5.3 Coppersmith-style attacks

5.3.1 M-SIDH

Let $p = ABf - 1$ such that $A = \prod_{i=1}^{t} r_i$ and $B = \prod_{i=1}^{t} q_i$ are co-prime integers, r_i, q_i are distinct small primes. In the M-SIDH protocol as described in 4.2, the scalar α

which masks the torsion point images is sampled from $\{x \in \mathbb{Z}/B\mathbb{Z} \mid x^2 = 1 \bmod B\}$. To break the scheme, one needs to recover α. Assuming the most significant bits (MSBs) of α leak, we can recover α using the original Coppersmith's method (as described in Section 3.1.1) for the polynomial equation $x^2 = 1 \bmod B$. For the sake of readability we re-iterate Theorem 3.1.2:

Theorem 5.3.1. *Let N be an integer of unknown factorization. Let $f(x)$ be a univariate monic polynomial of constant degree δ. Then we can find all solutions x_0 of the equation*

$$f(x) = 0 \bmod N \text{ with } |x_0| \leq N^{1/\delta}$$

in time polynomial of $\log N$ and δ for any $\varepsilon > 0$.

Assume access to the oracle $O(P) = ([\alpha]\varphi(P), \mathrm{MSB}_\ell(\alpha))$ which outputs the μ -MSBs of the scalar, then we can write

$$\alpha = A_0 + x_0$$

where A_0 is known and x_0 is unknown. We have the modular polynomial equation $p(x) = (A_0 + x)^2 - 1 = x^2 + 2A_0x + ({A_0}^2 + 1)$. Also, $|x_0| \leq \sqrt{B} - (A_0 + 1) < \sqrt{B} - A_0$. Define $X = \sqrt{B} - A_0$. Since μ MSBs are known, $|x_0| < 2^{k-\mu}$, where $k = \log_2(\alpha) \approx \log_2(B)$. To satisfy Theorem 3.1.2, we must have the bound X on the unknown x_0 such that $X < B^{\frac{1}{2}}$. Then we have that for $\mu = \frac{1}{2}\log_2 B$ and X, $|x_0| \leq 2^{\log_2(B) - \frac{1}{2}\log_2 B} = 2^{\frac{1}{2}\log_2 B}$. The following corollary then satisfies Theorem 3.1.2.

Corollary 5.3.2. *We can find α in time polynomial of $(\log B, 2^\delta)$ if we know $\frac{1}{2}\log_2 B$ most significant bits of α.*

5.3.2 FESTA

In the case of FESTA (as described in Section 4.3) the equation scalars obeying the equation $f(\alpha, \beta) = \alpha\beta - 1 = 0 \bmod p^k$ lends itself to Coppersmith-style approaches (as outlined in Sections 3.1.3 and 3.2). It must be noted that the case when the prime is a power of two, this method is superseded by the combinatorial attack (in Section 5.4), hence we focus on $p > 2$-powers in this section.

Case 1: shifting Blocks of Missing Bits

Assume that we have a leakage of μ-MSBs A and B of α and β, given by some oracle $O(\alpha, \beta)$. We write then

$$\alpha = A \cdot 2^{k-\mu} + x, \qquad \beta = B \cdot 2^{k-\mu} + y$$

such that $x, y \leq 2^{k-\mu}$ unknown and substitute that into $f(\alpha, \beta) = 0 \bmod p^k$. However, we can do the same thing if we additionally know the ℓ-LSBs too:

$$\alpha = A \cdot 2^{k-\mu} + x \cdot 2^{\ell} + a, \qquad \beta = B \cdot 2^{k-\mu} + y \cdot 2^{\ell} + b, \qquad x, y \leq 2^{k-\mu-\ell}$$

where a and b denote the known LSBs. Then the challenge towards solving Problem 5.2.2 in this scenario is to figure out the missing block of bits in between the leaked MSBs and LSBs.

By substituting the above equation into $f(a, b) = 0 \bmod p^k$ and ensuring that the coefficient of the leading monomial is 1, we can in fact recover an arbitrary block of consecutive missing bits, using the Automated Coppersmith with the polynomial $f(a, b) = 0 \bmod p^k$ using Theorem 3.2.4, and corresponding Algorithms 4, and parameter choices in Section 3.2.2. We re-iterate the theorem for readability:

Theorem 5.3.3. *Suppose we are given a modulus $M \in \mathbb{N}$, polynomial $f \in \mathbb{Z}_M[x_1, \ldots, x_k]$ and bounds $0 \leq X_1, \ldots, X_k \leq M$, where $k = \mathcal{O}(1)$. Furthermore, suppose we are given an integer $m \in N$, a set of monomials $\mathcal{M}$, a monomial order $\prec$ on $\mathcal{M}$ and an $(M, \prec)$-suitable set of polynomials $\mathcal{F} \subset \mathbb{Z}_{M^m}[x_1, \ldots, x_k]$ with*

$$Z_{M,X_1,\ldots,X_k}(f) \subseteq Z_{M^m,X_1,\ldots,X_k}(\mathcal{F}) \tag{5.1}$$

If the conditions

$$\prod_{\lambda \in \mathcal{M}} |\mathsf{LC}(\mathcal{F}[\lambda])| \leq \frac{M^{(m-k)|\mathcal{M}|}}{\prod_{\lambda \in \mathcal{M}} \lambda(X_1, \ldots, X_k)} \tag{5.2}$$

$\log(M) \geq |\mathcal{M} \geq m$ *and* $|\mathcal{M}| \geq k$ *hold, then we can compute all* $r \in Z_{M,X_1,\ldots,X_k}(f)$ *in time polynomial in* $\deg(\mathcal{F}) \cdot \log(M)$ *under Heuristic 3.2.2 for* $k > 1$.

One must note the reason why this approach only works for $p > 2$, since if $p = 2$ then the coefficient of the leading monomial is likely to contain a power of 2 and thus is not invertible. Repeating multiple experiments in SageMath [S+25], using the implementation available in [MN23], we note that asymptotically we can only expect to recover 25% of α and β with this method since for bounds X, Y on x, y we need to satisfy

$$XY < \left(p^k\right)^{1/\deg(f)} \qquad \Longrightarrow \qquad X, Y < p^{k/4}.$$

Case 2: missing Bits Scattered Around α

As mentioned above, it might be problematic to apply the above approach if we are getting random bits of α. However, it might be possible to write

$$\alpha = A_1 \cdot 2^{k-\mu_1} + x_1 \cdot 2^{\ell_1} + A_2 \cdot 2^{k-\mu_2} + x_2 \cdot 2^{\ell_2} \ldots$$

and similarly

$$\beta = B_1 \cdot 2^{k-\mu_1} + y_1 \cdot 2^{\ell_1} + \dots$$

where the $x_i, y_i \in \{0,1\}$ are (extremely) small roots of the multivariate polynomial $f(a,b) = ab - 1 \bmod p^k$. Assuming we are missing λ bits in each α and β (i.e. after substitution f has 2λ variables and total degree 2), then Coppersmith's condition becomes

$$\prod X_i Y_i < \left(p^k\right)^{1/2} \qquad \Longrightarrow \qquad X_i, Y_i < p^{k/4\lambda}.$$

As long as $p^{k/4\lambda} > 1$ this method still works as the roots are in $\{0,1\}$. However, since the runtime of all Coppersmith-style procedures are exponential in λ due to the fact that $\lambda \in \mathcal{O}(1)$. It is still in speculation if a sub-exponential attack might be developed if $\lambda \in \mathcal{O}(\sqrt{k})$, in such a scenario.

5.3.3 POKÉ

For POKÉ protocol (as described in Section 4.4), we have the parameters: prime p with the structure $p + 1 = 2^a 3^b 5^c f$ where $2^a \approx 2^\lambda$, $3^b \approx 2^{2\lambda}$ and $5^c \approx 2^{\lambda/3}$. We have the public key which contains the following torsion point information:

$$E_0, \quad E_1, \qquad (P_0, Q_0) \in E_0[2^a], \qquad (R_0, S_0) \in E_0[3^b], \qquad (X_0, Y_0) \in E_0[5^c]$$
$$(P_1, Q_1) = ([\alpha_2]\varphi_A(P_0), [\beta_2]\varphi_A(Q_0)), \quad (R_1, S_1) = ([\gamma_3]\varphi_A(R_0), [\gamma_3]\varphi_A(S_0)),$$
$$(X_1, Y_1) = ([\delta_5]\varphi_A(X_0), [\delta_5]\varphi_A(Y_0))$$

Using the properties of the Weil pairing (Cor. 2.1.21) on R_1, S_1 and X_1, Y_1 we can get the equations involving unknowns γ_3, δ_5 q

$$\gamma_3^2 q(2^a - q) = 1 \bmod 3^b, \qquad \delta_5^2 q(2^a - q) = 1 \bmod 5^c. \tag{5.3}$$

From the bilinearity of the pairing we can also get

$$\begin{aligned}
&e_{3^b5^c}(R_1 + X_1, S_1 + Y_1) \\
&= e(R_1, S_1) \cdot e(R_1, Y_1) \cdot e(X_1, S_1) \cdot e(X_1, Y_1) \\
&= e(R_0, S_0)^{\gamma_3^2 q(2^a - q)} \cdot e(R_0, Y_0)^{\gamma_3\delta_5 q(2^a - q)} \cdot e(X_0, S_0)^{\gamma_3\delta_5 q(2^a - q)} \cdot e(X_0, Y_0)^{\delta_5^2 q(2^a - q)} \\
&= \zeta^{\mu_1\gamma_3^2 q(2^a - q)} \cdot \zeta^{\mu_2\gamma_3\delta_5 q(2^a - q)} \cdot \zeta^{\mu_3\gamma_3\delta_5 q(2^a - q)} \cdot \zeta^{\mu_4\delta_5^2 q(2^a - q)}
\end{aligned}$$

where ζ is a generator of the group of 3^b5^c-th roots of unity and the exponents μ_i are known. Hence we also get the equation

$$(\mu_1\gamma_3^2 + (\mu_2 + \mu_3)\gamma_3\delta_5 + \mu_4\delta_5^2)q(2^a - q) = 1 \bmod 3^b5^c \tag{5.4}$$

for some known constants μ_i. Thus using Equations 5.3,5.4 we can use Automated Coppersmith 5.3.3, as in the previous section and recover γ_3, δ_5 and q. In accordance with the experiments, we observe that we require $> 80\%$ leakage for the scalars to be recovered by this method. This approach is a bit more heavy handed and slower here, since the equations involve quadratic here, in contrast to the linear equations used in Sections 5.3.1, 5.3.2. It is to be noted, that we break the POKÉ protocol if we can recover q itself, since the security of POKÉ relies heavily on the secrecy of the degree $q(2^a - q)$.

5.4 Combinatorial attacks

In this section we outline a combinatorial attacks against Problem 5.2.2, which outperform the Coppersmith-style attacks, in the case when we consider scalars modulo power of 2.

5.4.1 Procedure

A basic combinatorial approach trying to attack is the following: write scalars α, β symbolically as $2^i x_i$ and $2^i y_i$ respectively, where x_i is the i-th binary digit if known, and an unknown otherwise. We then compute the product $\alpha\beta$, which we know to be 1. For each $i \leq n$, where n is the number of bits of α and β, we obtain an equation $\mod 2^i$ including (possibly) some of the x_j, y_j. Notice that the variable x_j, if defined, will not appear in any equation $\mod 2^i$ with $i \leq j$, and will necessarily appear multiplied by 2^j in the equation $\mod 2^{j+1}$, since β is odd. The same is true for y_j.

This suggests to solve the equations incrementally, and keep track of the possible solutions of the variables involved. In this way, when we try to solve the equation $\mod 2^{i+1}$, we can evaluate all the variables x_j, y_j for $j < i$ in each of the possible solution we have kept so far. There are three possibilities:

1. Only one bit among x_i, y_i is known. In this case we have one equation and one binary variable, which admits a single solution. Thus for every valid evaluation of the variables up to $i-1$ we have exactly one valid evaluation of the variables up to i. We denote this situation as mismatch;

2. Neither x_i nor y_i are known. We have one equation in two binary variables, namely $x_i + y_i = c$ where c is 0 or 1 depending on the value assigned to the other variables. For each value of c, we have two possible combinations of x_i

and y_i satisfying the equation. Hence, the number of possible evaluations gets doubled. We call this situation a negative match;

3. Both x_y and y_i are known. We refer as this case as positive match. In this case we obtain one equation and zero variables. The fact that the equation $\alpha\beta = 1$ is satisfied for an evaluation when considered $\mod 2^i$ does not imply that the same evaluation multiplies to 1 $\mod 2^{i+1}$. On average, we expect this to happen roughly half of the times, and consequently to half the number of valid evaluations in this step. However, unlike in the other cases, this reasoning is only heuristic.

Leakage scenarios

I. In the situation where there are no leaks, we will encounter n negative matches, producing 2^n possible solutions as expected.

II. If, the first half of α and the second half of β are leaked, then we only encounter mismatches, and since we start with one possible solution we will only have one possible solution throughout the whole process and end up recovering α and β.

III. If otherwise both α and β leak their least significant bits, we will have first $n/2$ positive matches, leaving us with one (trivial) evaluation on 0 variables. After that, we will encounter $n/2$ negative matches, creating $2^{n/2}$ total possible evaluations.

5.4.2 Complexity

The complexity of this above procedure is given by the number of possible evaluations that we have to process. On average, given uniform leakages of half of the n bits of α and β, we expect $n/2$ mismatches, $n/4$ positive matches and $n/4$ negative matches. Notice, by definition, we have the number of positive matches equals the number of negative matches.

The actual complexity hence lies somewhere between the extreme case in which the positive matches have no effect, and there are $2^{n/4}$ possible solutions, and the extreme opposite in which the matches eat each other and we are never left with more than 2 solutions.

To get a better estimate of the concrete behavior of the algorithm, we run it for different bit sizes, with half of the bits uniformly leaked, and reported the average path width (i.e. the average number of valid evaluations of the variables through all the steps of the algorithm) computed over 1000 runs for each size. The results are shown

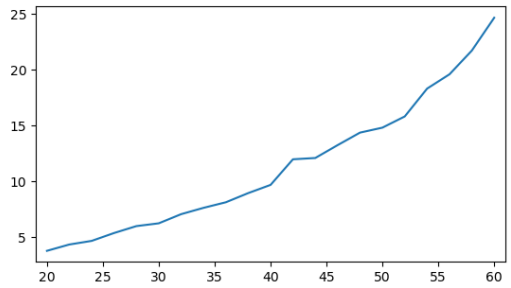


Figure 5.1: Average path width

in Figure 5.1. While probably still exponential, the performance of the algorithm is quite good. For comparison, the path width for $n = 60$ in the worst case, in which we have 15 positive matches followed by 15 negative ones and then 30 mismatches would be greater than 8700 instead of the observed average below 25. In practice, we can solve instances of 300 bits in minutes. This is indeed better in performance than the brute-forced Coppersmith method, and does not involve complicated fine tuning of parameters.

In conclusion, the complexity of the algorithm is upper bounded by $\mathcal{O}(nS)$ where S is the biggest size of Q_k during the execution, and Q_k is the list of pairs (x, y) such that $xy = 1 \mod 2^n$ with x, y matching known bits of α, β respectively, up to 2^k.

Chapter 6

On the choice of matrices made in FESTA

In this chapter, we reproduce the contents of the article [Das25].

6.1 Introduction

In the article [BMP23] the authors claim that the choice of diagonal matrices to scale torsion point images in the countermeasure FESTA is not a singular choice, and that the security of the scheme shall not be jeopardized if the commutative subgroup of diagonal matrices could be replaced by any other commutative subgroup of invertible matrices, such as that of circulant matrices[1]. In the framework of [FFP24], it is interesting to ask if the corresponding level structures reduce to each other. Here we confirm that the circulant case indeed reduces to the diagonal case as proposed in [BMP23] when the scaling matrices are defined over $(\mathbb{Z}/N\mathbb{Z})^\times$ for $N = p^r$ for prime $p > 2$. In the special case when the matrices are defined over finite fields i.e, $N = p$ for some large prime, the reduction to the diagonal case holds for any (non-trivial) commutative subalgebra. However, when $N = 2^k$, we show that a reduction between the two cases is not possible by our method, which is in contrast to the aforementioned claim.

[1]Implicitly, the choice of scaling matrices sourced from the group must be non-trivial to prevent exploitation by the SIDH attacks.

6.2 Preliminaries

6.2.1 Matrices

Definition 6.2.1. A $n \times n$ circulant matrix C takes the following form:

$$\begin{pmatrix} c_0 & c_{n-1} & \cdots & c_2 & c_1 \\ c_1 & c_0 & c_{n-1} & & c_2 \\ \vdots & c_1 & c_0 & \ddots & \vdots \\ c_{n-2} & & \ddots & \ddots & c_{n-1} \\ c_{n-1} & c_{n-2} & \cdots & c_1 & c_0 \end{pmatrix}$$

Definition 6.2.2. For a ring R and $n \geq 1$, let $\alpha \in R$ be a principal n-th root of unity. The Discrete Fourier transform over a ring R is defined as follows (in matrix notation):

$$\begin{pmatrix} f_0 \\ f_1 \\ \vdots \\ f_{n-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ 1 & \alpha^2 & \alpha^4 & \cdots & \alpha^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{n-1} & \alpha^{2(n-1)} & \cdots & \alpha^{(n-1)(n-1)} \end{pmatrix} \begin{pmatrix} v_0 \\ v_1 \\ \vdots \\ v_{n-1} \end{pmatrix}$$

6.2.2 Level structures

We shall use the framework of isogeny problems with level structure as proposed in [FFP24] to phrase the underlying problem in FESTA. The definition of a Γ-SIDH problem is as follows:

Definition 6.2.3. Fix coprime integers d, N and $\Gamma \leq GL_2(\mathbb{Z}/N\mathbb{Z})$. Let $E \xrightarrow{\phi} E'$ be an isogeny of degree d and S be a Γ level structure.
The (d, Γ)-modular isogeny problem (of level N) asks that given $(E, S, E', \phi(S))$ to compute ϕ. When d is clear, this is referred to as the Γ-SIDH problem.

If one replaces Γ by $\left\{ \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix} \right\} \leq SL_2$, then we have the underlying Γ-SIDH problem for FESTA, and analogously for other Γ.

6.3 Reduction

Lemma 6.3.1. *For a matrix $A \in SL_2(\mathbb{Z}/N\mathbb{Z})$ and $\Gamma \leq SL_2(\mathbb{Z}/N\mathbb{Z})$, a Γ-SIDH problem reduces to $A^{-1}\Gamma A$-SIDH problem, given an oracle to solve discrete log in*

$\mu_N \subset \mathbb{F}_{q^r}^{\times}$, *the subgroup of* n*th roots of unity.*

Proof. Let (E, S, E', S') be a Γ-SIDH problem. Choose a representative (P, Q) of S, and compute its Weil pairing $W_1 := e_N(P, Q)$. Define $\bar{S} = A^{-1}\Gamma A \cdot (P, Q)$. The Weil pairing gives us

$$e_N(\phi(\bar{S})) = e_N(\bar{S})^{\deg \phi} = W_1^d$$

Now, choose a representative $(P', Q') := \phi(P, Q)$ of $\phi(S)$ and compute the Weil pairing $W_2 = e_N(P', Q')$. Use the oracle to compute discrete logarithm x of W_1^d to base W_2 and find a matrix $\gamma' \in \Gamma$ such that $\det \gamma' = x$. Define $\bar{S}' := A^{-1}\Gamma A \cdot \gamma' \cdot (P', Q')$; then $\bar{S}' = \phi(\bar{S})$. Hence $(E, \bar{S}, E', \bar{S}')$ is an instance of $A^{-1}\Gamma A$-SIDH problem, having the same solution as the Γ-SIDH problem. □

6.3.1 For $N = p^k$ with odd p

Lemma 6.3.2. *If* C *denotes a circulant matrix defined over*[2] $\mathbb{Z}/N\mathbb{Z}$ *and* F *denotes the Discrete Fourier transform matrix defined over* $\mathbb{Z}/N\mathbb{Z}$*, then for some diagonal matrix* D *we have that* $C = F^{-1}DF$*.*

Proof. Any circulant matrix can be decomposed into a polynomial in terms of the permutation matrix P as $C = \sum_{i=0}^{n} c_i P^i$ where c_i are entries of the circulant matrix. Since the permutation matrix is defined over R, the eigenvectors of P are $(\alpha^k, \alpha^{2k}, \ldots, \alpha^{(n-1)k})$ for $0 \leq k \leq n-1$ where α is a principal n-th root of unity. Then the permutation matrix (and subsequently linear combination of it's powers) can be diagonalized by conjugating with a matrix which has the eigenvectors as columns. This matrix is precisely the Discrete Fourier Transform matrix as defined above. Hence the circulant matrix can be written as $C = F^{-1}DF$ where D is the diagonal matrix obtained from the linear combination of diagonal matrices. □

Using the above two lemmas, one can conclude the following theorem:

Theorem 6.3.3. *For* $\mathfrak{D} = \left\{ \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix} \right\}$ *and* $\mathfrak{C} = \left\{ \begin{pmatrix} a & b \\ b & a \end{pmatrix} \right\}$ *such that both are subgroups of* $SL_2(\mathbb{Z}/N\mathbb{Z})$*, where* $N = p^k$ *for* $p > 2$ *and* $k > 0$*. Then the* $\mathfrak{C}$*-SIDH problem reduces to* $\mathfrak{D}$*-SIDH problem.*

[2] This theorem holds for any ring R such that F is invertible in $\mathcal{M}_{n\times n}(R)$.

6.3.2 For $N = 2^k$

Theorem 6.3.4. *For a invertible matrix of the form* $\begin{pmatrix} a & b \\ b & a \end{pmatrix}$, *there does not exist a diagonalization over the ring* $\mathbb{Z}/N\mathbb{Z}$ *where* $N = 2^k$, *for* $k > 0$.

Proof. Let $A : \begin{pmatrix} a & b \\ b & a \end{pmatrix} \in SL_2(\mathbb{Z}/N\mathbb{Z})$ such that it is invertible, which implies $\det A = (a^2 - b^2)$ is a unit. Since the odd numbers in $(\mathbb{Z}/N\mathbb{Z})$ are the units, it is not possible if both a and b are odd (or even), hence one must be odd and the other must be a even for the matrix A to be invertible. Without loss of generality, assume that a is a even and b is a odd.

The characteristic polynomial of A is $p(t) = (t-a)^2 - b^2$. If we solve the equation for the eigenvalues, $(t-a)^2 = b^2 \bmod N$ entails that $(t-a)$ is a odd since b is a odd. Since a is a even and $(t-a)$ is a odd, it implies that the eigenvalues must be a odd. Let λ be an eigenvalue of A and $v := \begin{pmatrix} x \\ y \end{pmatrix}$ be the corresponding eigenvector. From the equation $Av = \lambda v$ we obtain the equations $ax + by = \lambda x \bmod N$ and $ay + bx = \lambda y \bmod N$. Adding both of them, we obtain $(a+b-\lambda)(x+y) = 0 \bmod N$.

Suppose x, y are not both odd (or even) at the same time, i.e, $(x+y)$ and $(x-y)$ are odd. This implies that $(a+b-\lambda) = 0 \bmod N \implies a+b = \lambda \bmod N$. Then substituting λ in the equation $ax + by = \lambda x \bmod N$ we have $b(y-x) = 0 \bmod N$. Since both are odd by assumption, it is a clear contradiction. Hence x, y must be both odd (or even). We have the modular matrix[3] (v_1, v_2) obtained from the corresponding eigenvectors v_i whence $v_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix}$ with x_i, y_i being both odd (or even). However for all possible combinations of v_i (i.e, when v_i is comprised of units or non-units), the modular matrix turns out to be singular. This entails that a diagonalization is not possible for A over the ring $(\mathbb{Z}/N\mathbb{Z})$. □

Hence our strategy of the previous section fails and we cannot say anything conclusively regarding the reduction of the circulant case to the original diagonal case. This is indeed contrasting to the claim of [BMP23], since this reduction does not hold when $N = 2^k$, a parameter choice made in FESTA.

[3]The matrix P such that $A = PDP^{-1}$ where D is a diagonal matrix.

6.3.3 Finite Fields

For a finite field $k = \mathbb{Z}/p\mathbb{Z}$ for $p > 2$, it is a well known result that the 2-dimensional commutative matrix subalgebras of $\mathcal{M}_{n\times n}(k)$ could be classified up to isomorphism as follows:

$$\mathfrak{D} = \left\{ \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix} \right\} \quad \mathfrak{C} = \left\{ \begin{pmatrix} a & b \\ b & a \end{pmatrix} \right\} \quad \mathfrak{T} = \left\{ \begin{pmatrix} a & b \\ 0 & c \end{pmatrix} \right\}$$

In [FFP24] the authors have already showed the reductions between $\mathfrak{T}$ and $\mathfrak{D}$. In Theorem 3.3 above, we have shown that $\mathfrak{C}$ reduces to $\mathfrak{D}$. Thus one can conclude that for $N = p$, the choice for any commutative subalgebra in FESTA still reduces to the original formulation of FESTA.

Bibliography

[BDD+24] Andrea Basso, Pierrick Dartois, Luca De Feo, Antonin Leroux, Luciano Maino, Giacomo Pope, Damien Robert, and Benjamin Wesolowski. SQIsign2D-west - the fast, the small, and the safer. In Kai-Min Chung and Yu Sasaki, editors, ASIACRYPT 2024, Part III, volume 15486 of LNCS, pages 339–370. Springer, Singapore, December 2024. doi:10.1007/978-981-96-0891-1_11.

[BKM+21] Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Antonio Sanso. Cryptanalysis of an oblivious PRF from supersingular isogenies. In Mehdi Tibouchi and Huaxiong Wang, editors, ASIACRYPT 2021, Part I, volume 13090 of LNCS, pages 160–184. Springer, Cham, December 2021. doi:10.1007/978-3-030-92062-3_6.

[BM24] Andrea Basso and Luciano Maino. POKÉ: A compact and efficient PKE from higher-dimensional isogenies. Cryptology ePrint Archive, Paper 2024/624, 2024. URL: https://eprint.iacr.org/2024/624.

[BMP23] Andrea Basso, Luciano Maino, and Giacomo Pope. FESTA: Fast encryption from supersingular torsion attacks. In Jian Guo and Ron Steinfeld, editors, ASIACRYPT 2023, Part VII, volume 14444 of LNCS, pages 98–126. Springer, Singapore, December 2023. doi:10.1007/978-981-99-8739-9_4.

[Buc70] B. Buchberger. Ein algorithmisches kriterium für die lösbarkeit eines algebraischen gleichungssytems. Aequationes mathematicae, 4(3):374–383, 1970.

[Buc76] B. Buchberger. Theoretical basis for the reduction of polynomials to canonical forms. ACM SIGSAM Bulletin, 10(3):19–29, 1976.

[CD22] Wouter Castryck and Thomas Decru. An efficient key recovery attack on SIDH (preliminary version). Cryptology ePrint Archive, Report 2022/975, 2022. URL: https://eprint.iacr.org/2022/975.

[Cop96a] Don Coppersmith. Finding a small root of a bivariate integer equation; factoring with high bits known. In Ueli M. Maurer, editor, EUROCRYPT'96, volume 1070 of LNCS, pages 178–189. Springer, Berlin, Heidelberg, May 1996. doi:10.1007/3-540-68339-9_16.

[Cop96b] Don Coppersmith. Finding a small root of a univariate modular equation. In Ueli M. Maurer, editor, EUROCRYPT'96, volume 1070 of LNCS, pages 155–165. Springer, Berlin, Heidelberg, May 1996. doi:10.1007/3-540-68339-9_14.

[Cop97] Don Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. Journal of Cryptology, 10(4):233–260, September 1997. doi:10.1007/s001459900030.

[Cor04] Jean-Sébastien Coron. Finding small roots of bivariate integer polynomial equations revisited. In Christian Cachin and Jan Camenisch, editors, EUROCRYPT 2004, volume 3027 of LNCS, pages 492–505. Springer, Berlin, Heidelberg, May 2004. doi:10.1007/978-3-540-24676-3_29.

[Cou06] Jean-Marc Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291, 2006. URL: https://eprint.iacr.org/2006/291.

[CV23] Wouter Castryck and Frederik Vercauteren. A polynomial time attack on instances of M-SIDH and FESTA. In Jian Guo and Ron Steinfeld, editors, ASIACRYPT 2023, Part VII, volume 14444 of LNCS, pages 127–156. Springer, Singapore, December 2023. doi:10.1007/978-981-99-8739-9_5.

[Das25] Subham Das. A Note on FESTA, 2025. URL: http://cryptosubh.github.io/assets/notefesta.pdf.

[De 17] Luca De Feo. Mathematics of isogeny based cryptography. CoRR, abs/1711.04062, 2017. URL: http://arxiv.org/abs/1711.04062, arXiv:1711.04062.

[De 20] Luca De Feo. Tools for designing protocols based on isogenies, 2020. URL: https://defeo.lu/docet/assets/misc/2021-08-02-isogeny-school.pdf.

[DH76] W. Diffie and M. Hellman. New directions in cryptography. IEEE Transactions on Information Theory,, 22(6):644–654, 1976.

[DLLW23] Luca De Feo, Antonin Leroux, Patrick Longa, and Benjamin Wesolowski. New algorithms for the deuring correspondence - towards practical and secure SQISign signatures. In Carmit Hazay and Martijn Stam, editors, EUROCRYPT 2023, Part V, volume 14008 of LNCS, pages 659–690. Springer, Cham, April 2023. doi:10.1007/978-3-031-30589-4_23.

[dQKL+21] Victoria de Quehen, Péter Kutas, Chris Leonardi, Chloe Martindale, Lorenz Panny, Christophe Petit, and Katherine E. Stange. Improved torsion-point attacks on SIDH variants. In Tal Malkin and Chris Peikert, editors, CRYPTO 2021, Part III, volume 12827 of LNCS, pages 432–470, Virtual Event, August 2021. Springer, Cham. doi:10.1007/978-3-030-84252-9_15.

[Fau99] J.C. Faugére. A new efficient algorithm for computing gröbner bases. Journal of Pure and Applied Algebra, 139:61–88, 1999.

[FFP24] Luca De Feo, Tako Boris Fouotsa, and Lorenz Panny. Isogeny problems with level structure. Cryptology ePrint Archive, Paper 2024/459, 2024. URL: https://eprint.iacr.org/2024/459.

[FMP23] Tako Boris Fouotsa, Tomoki Moriya, and Christophe Petit. M-SIDH and MD-SIDH: Countering SIDH attacks by masking information. In Carmit Hazay and Martijn Stam, editors, EUROCRYPT 2023, Part V, volume 14008 of LNCS, pages 282–309. Springer, Cham, April 2023. doi:10.1007/978-3-031-30589-4_10.

[FP22] Tako Boris Fouotsa and Christophe Petit. A new adaptive attack on SIDH. In Steven D. Galbraith, editor, CT-RSA 2022, volume 13161 of LNCS, pages 322–344. Springer, Cham, March 2022. doi:10.1007/978-3-030-95312-6_14.

[Gal12] Steven Galbraith. Mathematics of Public Key Cryptography. Cambridge University Press, London, 2012.

[Gal22] Steven Galbraith. Attacks on sidh/sike, 2022. URL: https://ellipticnews.wordpress.com/2022/08/12/attacks-on-sidh-sike/.

[HG97] N. Howgrave-Graham. Finding small roots of univariate modular equations revisited. Lecture Notes in Computer Science, 1335:131–142, 1997.

[HLWW13] Carmit Hazay, Adriana López-Alt, Hoeteck Wee, and Daniel Wichs. Leakage-resilient cryptography from minimal assumptions. In Thomas

Johansson and Phong Q. Nguyen, editors, EUROCRYPT 2013, volume 7881 of LNCS, pages 160–176. Springer, Berlin, Heidelberg, May 2013. doi:10.1007/978-3-642-38348-9_10.

[Jao22] David Jao. Supersingular isogeny key encapsulation, 2022. URL: https://csrc.nist.gov/csrc/media/Projects/post-quantum-cryptography/documents/round-4/submissions/SIKE-spec.pdf.

[JD11] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011, pages 19–34. Springer, Berlin, Heidelberg, November / December 2011. doi:10.1007/978-3-642-25405-5_2.

[Kan97] E. Kani. The number of curves of genus two with elliptic differentials. Journal für die reine und angewandte Mathematik, 485:93–122, 1997.

[KL20] Jonathan Katz and Yehuda Lindell. Introduction to Modern Cryptography, 3rd Edition. Chapman and Hall, London, 2020.

[May09] Alexander May. Using lll-reduction for solving rsa and factorization problems. In The LLL algorithm, pages 315–348. Springer, 2009.

[May21] Alexander May. Lattice-based integer factorization - an introduction to coppersmith's methods. In Computational Cryptography, number 4, pages 78–105. Cambridge University Press, 2021.

[MM22] Luciano Maino and Chloe Martindale. An attack on SIDH with arbitrary starting curve. Cryptology ePrint Archive, Report 2022/1026, 2022. URL: https://eprint.iacr.org/2022/1026.

[MN23] Jonas Meers and Julian Nowakowski. Solving the hidden number problem for CSIDH and CSURF via automated coppersmith. In Jian Guo and Ron Steinfeld, editors, ASIACRYPT 2023, Part IV, volume 14441 of LNCS, pages 39–71. Springer, Singapore, December 2023. doi:10.1007/978-981-99-8730-6_2.

[NO23] Kohei Nakagawa and Hiroshi Onuki. QFESTA: Efficient algorithms and parameters for FESTA using quaternion algebras. Cryptology ePrint Archive, Report 2023/1468, 2023. URL: https://eprint.iacr.org/2023/1468.

[NR19] Michael Naehrig and Joost Renes. Dual isogenies and their application to public-key compression for isogeny-based cryptography. In Steven D. Galbraith and Shiho Moriai, editors, ASIACRYPT 2019, Part II, volume 11922 of LNCS, pages 243–272. Springer, Cham, December 2019. doi:10.1007/978-3-030-34621-8_9.

[Pan22] Lorenz Panny. You could have broken sidh, 2022. URL: https://yx7.cc/blah/2022-08-22.html.

[Pet17] Christophe Petit. Faster algorithms for isogeny problems using torsion point images. In Tsuyoshi Takagi and Thomas Peyrin, editors, ASIACRYPT 2017, Part II, volume 10625 of LNCS, pages 330–353. Springer, Cham, December 2017. doi:10.1007/978-3-319-70697-9_12.

[Rob22] Damien Robert. Breaking SIDH in polynomial time. Cryptology ePrint Archive, Report 2022/1038, 2022. URL: https://eprint.iacr.org/2022/1038.

[Rob24] Damien Robert. On the efficient representation of isogenies (a survey). Cryptology ePrint Archive, Report 2024/1071, 2024. URL: https://eprint.iacr.org/2024/1071.

[S+25] W. A. Stein et al. Sage Mathematics Software (Version x.y.z). The Sage Development Team, 2025. http://www.sagemath.org.

[Sil92] Joseph H. Silverman. The arithmetic of elliptic curves, volume 106 of Graduate Texts in Mathematics. Springer-Verlag, New York, 1992.

[Tat66] J. Tate. Endomorphisms of abelian varieties over finite fields. Inventiones mathematicae, 2:134–144, 1966. URL: http://eudml.org/doc/141848.

[V9char"03017] J. Vélu. Isogénies entre courbes elliptiques. J. R. Acad. Sci. Paris Sér., A-B 273:A238–A241, 1997.